# NAVAL POSTGRADUATE SCHOOL

Monterey, California

# THESIS

## COMPUTER SIMULATED DEVELOPMENT OF IMPROVED COMMAND TO LINE-OF-SIGHT MISSILE GUIDANCE TECHNIQUES

by

Frank F. Hewitt

March 1979

Thesis Advisor: H. A. Titus

Approved for public release; distribution unlimited.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

**REPORT DOCUMENTATION PAGE**

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)

6 Computer Simulated Development of Improved Command to Line-of-Sight Missile Guidance Techniques .

5. ~~TYPE OF REPORT & PERIOD COVERED~~

9 Master's Thesis,
~~March 1979~~

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)

10 Frank F. Hewitt

8. CONTRACT OR GRANT NUMBER(s)

12 107p.

9. PERFORMING ORGANIZATION NAME AND ADDRESS

Naval Postgraduate School
Monterey, CA 93940

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS

Naval Postgraduate School
Monterey, CA 93940

12. ~~REPORT DATE~~

11 March 1979

13. ~~NUMBER OF PAGES~~

106

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

Naval Postgraduate School
Monterey, CA 93940

15. SECURITY CLASS. (of this report)

Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Line-of-sight Guidance
Digital missile simulation
Missile guidance

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

→ Three computer simulations of command to line-of-sight missile guidance systems were developed within an electronic warfare environment to test the feasibility of improved guidance techniques while maintaining simplicity for generic application among the various missiles of this general guidance classification. The first simulation modeled the basic guidance scheme; the second introduced a "lead-angle" —> next page

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

251450

Cond.

concept; and the third simulation combined these techniques for use depending upon whether the target took evasive action or employed electronic countermeasures. It was found that consideration should be given for use of a "lead-angle" variant in conjunction with the basic guidance technique to enhance the effective engagement envelope of these missile systems against relatively slower-maneuvering targets employing only low-duty-cycle denial jamming.

Approved for public release; distribution unlimited

# COMPUTER SIMULATED DEVELOPMENT
# OF IMPROVED
# COMMAND TO LINE-OF-SIGHT MISSILE GUIDANCE TECHNIQUES

by

Frank F. Hewitt
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1967

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS TECHNOLOGY

from the

NAVAL POSTGRADUATE SCHOOL
March 1979

Author: ______________________

Approved by: ______________________
Thesis Advisor

David B. Hoisington
Second Reader

David B. Hoisington
Chairman, Electronic Warfare Academic Group

______________________
Academic Dean

# ABSTRACT

Three computer simulations of command to line-of-sight missile guidance systems were developed within an electronic warfare environment to test the feasibility of improved guidance techniques while maintaining simplicity for generic application among the various missiles of this general guidance classification. The first simulation modeled the basic guidance scheme; the second introduced a "lead-angle" concept; and the third simulation combined these techniques for use depending upon whether the target took evasive action or employed electronic countermeasures. It was found that consideration should be given for use of a "lead-angle" variant in conjunction with the basic guidance technique to enhance the effective engagement envelope of these missile systems against relatively slower-maneuvering targets employing only low-duty-cycle denial jamming.

## TABLE OF CONTENTS

# LIST OF FIGURES

FIGURE PAGE

## TABLE OF VARIABLES

| DIAGRAM VARIABLES | COMPUTER VARIABLES | NOUN DESCRIPTION |
|---|---|---|
| CRE | X(1) | cross-range error |
| EX | EX | error function |
| G | G | In the 'consolidated' simulation this is the magnitude of missile's Y-direction acceleration. In the other simulations this is tne missile's cross-range acceleration. |
| LOS | SIGT | line of sight to target |
| $R_m$ | RM | missile range from tracker |
| SLOS | SLOS | synthetic line of sight |
| $\dot{\text{SLOS}}$ | SLOSD | rate of synthetic line-of-sight's movement |
| t | T | time |
| U | U | In the 'consolidated' simulation this is the missile's Y-direction acceleration. In the other simulations this is the missile's cross-range acceleration. |
| $X_i$ | X(18) | impact point's "X" coordinate |
| $\dot{X}_i$ | VIX | X-direction velocity of impact point |
| $Y_i$ | X(17) | impact point's "Y" coordinate |
| $\dot{Y}_i$ | VIY | Y-direction velocity of impact point |
| $X_m$ | X(8) | missile's "X" coordinate |
| $\dot{X}_m$ | VMX | X-direction velocity of missile |
| $\ddot{X}_m$ | VMXD | X-direction acceleration of missile |

| DIAGRAM VARIABLES | COMPUTER VARIABLES | NOUN DESCRIPTION |
|---|---|---|
| $X_t$ | X(28) | target's "X" coordinate |
| $\dot{X}_t$ | VTX | X-direction velocity of target |
| $\ddot{X}_t$ | VTXD | X-direction acceleration of target |
| $Y_m$ | X(7) | missile's "Y" coordinate |
| $\dot{Y}_m$ | VMY | Y-direction velocity of missile |
| $\ddot{Y}_m$ | VMYD | Y-direction acceleration of missile |
| $Y_t$ | X(27) | target's "Y" coordinate |
| $\dot{Y}_t$ | VTY | Y-direction velocity of target |
| $\ddot{Y}_t$ | VTYD | Y-direction acceleration of target |
| $\varepsilon$ | E | angular error |
| $\dot{\varepsilon}$ | EDOT | angular error rate |
| $\sigma_m$ | SIGM | angle of missile from reference |
| $\dot{\sigma}_m$ | SIGMD | angular rate of missile movement |
| $\sigma_t$ | SIGT | angle of target from reference |
| $\dot{\sigma}_t$ | SIGTD | angular rate of target movement |
| 1/s | (none) | integration |

# I. INTRODUCTION

Command to line-of-sight guidance for several types of surface-to-air missiles was developed through computer simulation. Consideration was given to electronic countermeasures against the guidance methods of this type.

The simulations are of sufficient complexity to demonstrate feasibility while maintaining simplicity for generic application among the various missiles of this general guidance classification.

# II. OVER-VIEW OF GUIDANCE TYPES

Guidance systems can be considered to fall, generally, into one of three basic types: line of sight, pursuit, or proportional navigation.

The line-of-sight system maintains the missile on the vector from the ground-based point of missile control (the tracker) to the target aircraft. Variations to this system can provide for "lead-angle" intercept through missile command computer generation of a synthetic line of sight upon which the missile is commanded to fly for target intercept. Also, beam-riding missiles fit this category. However, in the beam-riding type of line-of-sight guidance the missile avionics are more complicated and expensive because the missile normally senses its position relative to the beam center and generates its own guidance commands internally. This extends the missile's conceivable range but usually necessitates some form of terminal homing. Basic command to line-of-sight systems can utilize relatively cheap missiles by performing guidance computations in the tracker unit and not in the missile itself. This limits the missile to short-range scenarios and requires that the missile possess a greater speed advantage over the target.

Pursuit guidance systems maintain the missile's velocity vector toward the target. Generally, sensors in the missile can be assumed to perform this function; therefore, the missile can be forgotten by the launcher after the missile has locked onto the target. This type of system is capable of medium range work against a slowly-maneuvering target. It has relatively simple processing avionics.

Proportional navigation guidance systems hold a constant angle between the missile-to-target line of sight and the missile axis, thereby, generating a "constant-bearing-decreasing-range", collision situation. In other words, the change in missile heading is adjusted proportionally to the rate of change in the missile-to-target line of sight. This type of guidance is good for long ranges (and short) against highly-maneuvering targets. However, the missile requires either seeker and angle-rate sensing, or a command guidance signal based upon a collision course.

# III. TYPICAL ENGAGEMENT SEQUENCE

In order to provide a "vehicle" through which to better understand the basic aspects of command to line-of-sight guidance methodology, the engagement sequence of a short-range, air-defense, missile system is described. The Roland system was selected because the general operational aspects of the system are available at the unclassified level [1].

The entry of one or more aerial targets into the range of the search radar is indicated to the Roland vehicle commander by an audible tone. At the same time, a synthetic display of the targets appears on a screen to give the commander all the information needed to select the most threatening target. The screen images are different for friendly and enemy targets. Also, the entry of the target into the missile envelope, utilizing targét advanced-range computations, is indicated by a change in the display. With the search antenna raised and the search radar activated, target acquisition is possible even when the vehicle is in motion.

There are three modes of identification, friend or foe (IFF) interrogation: automatic, manual, and automatic within a given range.

When the commander has recognized a target as hostile and decided to engage it, he places a cursor over the screen image. This automatically brings the turret to bear and tracking can commence in either the "radar" or "optical" modes.

In the "radar" mode, the tracking radar automatically accepts target designation from the search radar, searches for, locks onto, and tracks the target.

In the "optical" mode, the aimer searches for the target in elevation with an optical sight. To aid him an electronic instrument displays the maximum theoretical elevation for the search. When the aimer has acquired the target in his cross-hairs, he keeps the target in his sight by manipulating a control stick. This control keeps the target properly positioned by moving the turret in azimuth and swivelling a mirror in elevation.

As soon as the commander confirms that the target is within missile range, he initiates the firing sequence in the "radar" mode, or authorizes "optical" mode firing through a command displayed in the aimer's sight. The aimer, then, can initiate the firing sequence.

The missile is guided by a command to line-of-sight technique. This means that the target is tracked optically or by radar and the deviation of the missile from this line of sight is determined and corrected by a guidance command. The commander may switch from "radar" to "optical" and back again, as desired, even after the missile has been launched.

Target tracking and determination of the missile's deviation from the line of sight are different for each mode. In the "radar" mode, the guidance radar has two receiving channels. One is used for target tracking and the other is used to locate the missile in the radar lobe through reception of the missile's radio frequency beacons. By comparing these angles, an error between the missile and the target line of sight can be determined. In the "optical" mode, a biaxially-stabilized mirror is manually controlled to keep the target vertically in the aimer's sight and the turret is rotated to the azimuth of the target line of sight. An infrared goniometer is mounted to provide missile angle from the tracker by following flares mounted on the rear of the missile. Then, a deviation of the missile angle from the target line of sight can be determined.

Two groups of signals are introduced into the command computer: the velocity of the line of sight in azimuth and elevation, and the deviation of the missile from the line of sight in azimuth and elevation. Based upon data from the line-of-sight movement and the angular deviations of the missile, the necessary guidance signals are calculated.

The guidance signals are relayed to the missile by a transmitter with highly directional characteristics. The command-transmitting antenna is slaved to the missile angle in both azimuth and elevation. It, therefore, is trained on the missile continuously.

The side forces required for missile course corrections are produced through deflection of the exhaust jet of the sustainer motor by spoilers at the rear of the missile (thrust-vector control).

When the missile reaches the point of impact with the target, the warhead is detonated by either percussion, contact fuse or the radio-frequency, proximity fuse. The warhead consists of a radial-effect, multiple-fragmentation charge.

Figure III-1 presents an operational schematic of the basic Roland missile system operation.

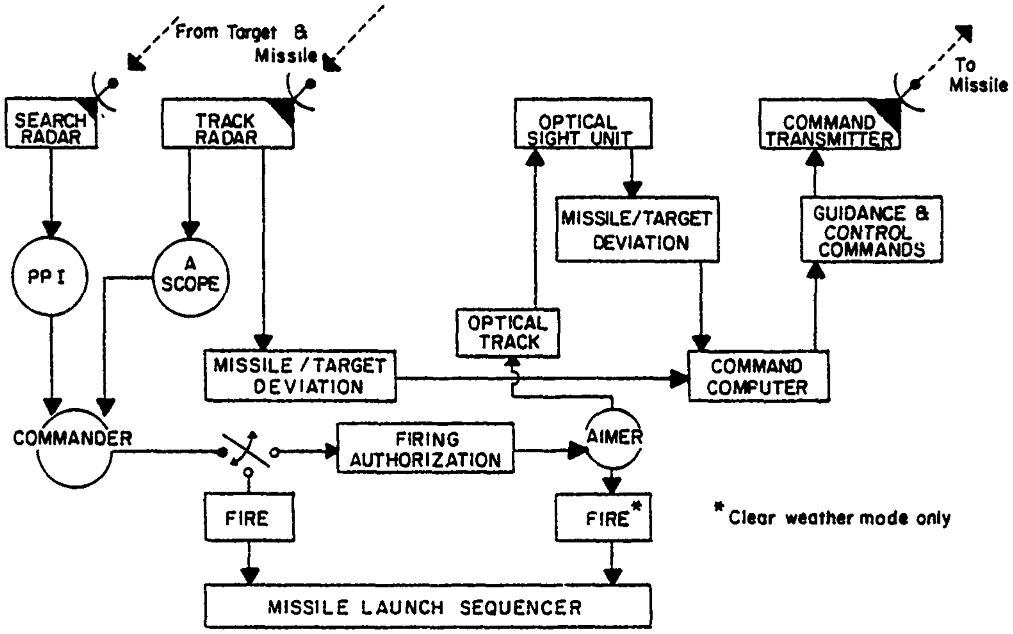


Figure III-1 Roland Missile System Operational Schematic

The computer simulations contained herein are generic in nature within the command to line-of-sight guided-missile type and have only reasonable estimates of missile capabilities introduced. This ensures unclassified results. At the same time, the simulations are of sufficient complexity to properly weigh the relative merits of the guidance variations discussed.

## IV. COMMAND TO LINE-OF-SIGHT GUIDANCE GEOMETRY

There are many tactical, inherently short-ranged, surface-to-air missiles which use this guidance technique for their operation. Its objective, basically, is to keep the error term ($\varepsilon$) zero so that the missile flies directly up the line of sight.

Since only angles to the target and to the missile are required for guidance, attempts to preclude target range through denial jamming prove useless against this type of guidance system. The only necessary information is, in general, to know whether the target is within the missile envelope. Deceptive down-link jamming, however, appears to be feasible. One possible method is presented in Section VI.

The geometry depicted in Figure IV-1 summarizes the "basic" command to line-of-sight geometric situation.

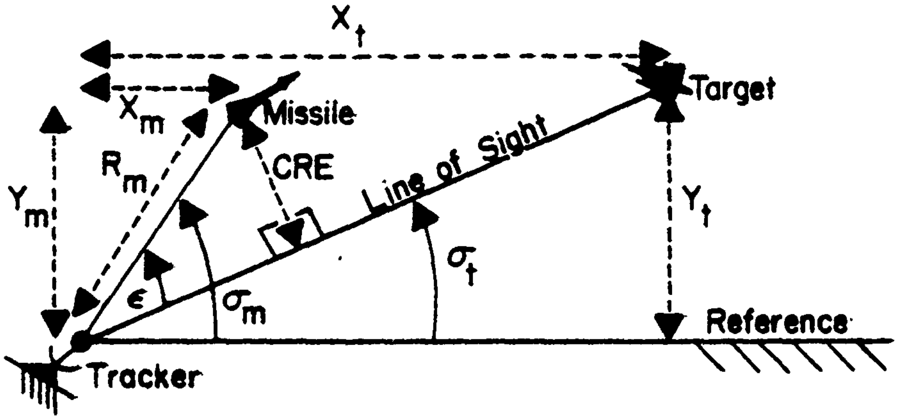


Figure IV-1 "Basic" Command to Line-of-Sight Geometry

A variant of this "basic" command to line-of-sight guidance approach, which shows promise for increasing system effectiveness, is to incorporate "lead-angle" into the basic guidance model. This technique would increase the effective range of the missile against a non-maneuvering, crossing target; however, it is degraded by a highly-maneuvering target. Since range information is required to calculate the impact point, additional electronic countermeasures could be employed by the target to deny this range information thus negating the "lead-angle" command to line-of-sight missile guidance scheme.

The previous "basic" geometry has been retained but it has been modified to utilize a synthetic line of sight upon which the missile flies. This synthetic line of sight is the direction of the impact point from the ground tracking unit. Therefore, the missile is commanded to fly toward the impact point rather than directly toward the target. This method, then, reduces the missile's flight time until impact, thus, improving the missile system's effective engagement range. Figure IV-2 shows this geometry.

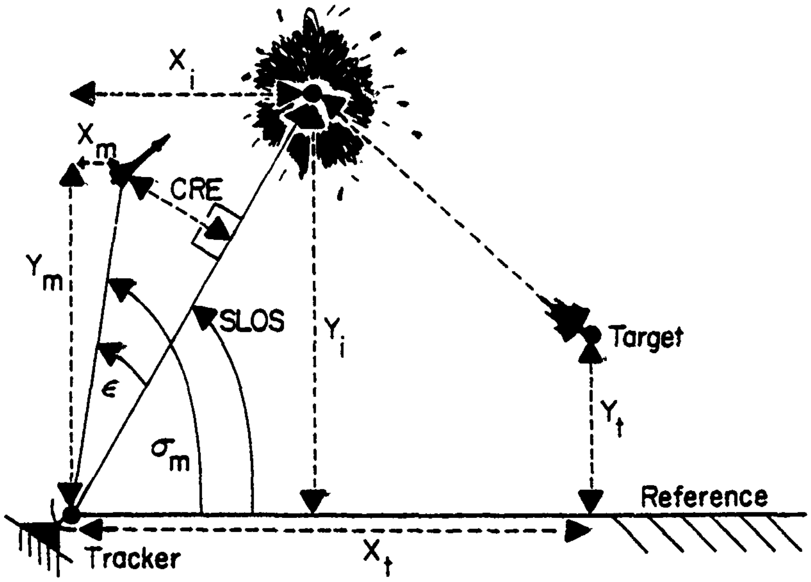


Figure IV-2 "Lead-Angle" Command to Line-of-Sight Geometry

# V. ON-OFF, THRUST-VECTOR, MISSILE CONTROL

The guidance models used in the computer simulations utilize on-off control. This type of control also is known as relay control or "bang-bang" control because the system's actuator is not moved in proportion to the error signal but, rather, is positioned either totally in the positive direction or totally in the negative direction. This causes the missile's thrust-vector control to apply missile-maneuvering thrust fully in one direction or the other. At first glance this type of system might appear to be very crude; however, it does have major advantages. First, the system does not require complex proportional amplification; the unit is smaller than its proportional control counterpart; it is, generally, the time-optimal control system; it weighs less than other systems; and, finally, it is relatively inexpensive [2].

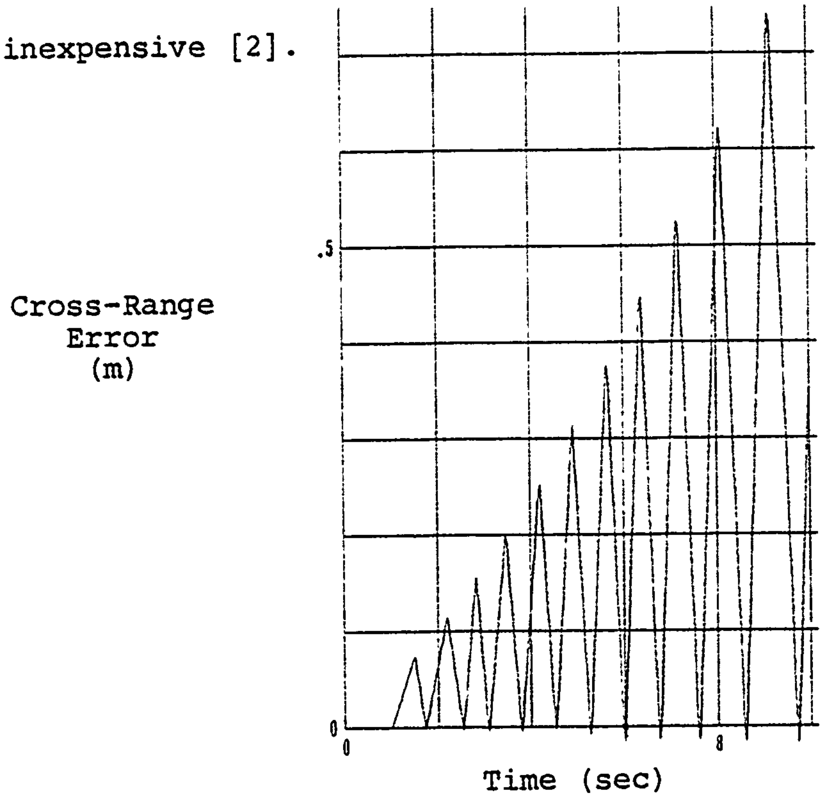


Figure V-1 Missile Cross-Range Error with Zero Initial Values for Cross-Range Error and Error Rate

Based upon this type of missile control system, initial computer simulations, not presented here in detail, were designed. They provided an elementary computer program structure which was modified incrementally to generate improved computer models. In these first programs, the error values were introduced directly into the missile's thrust-vector control without error rate. This produced increased missile cross-range error with flight time as shown in Figure V-1. This effect has been eliminated in the subsequent computer guidance models by proper acceleration-limiting techniques. Additionally, if initial erroneous values of cross-range error and cross-range error rate are introduced

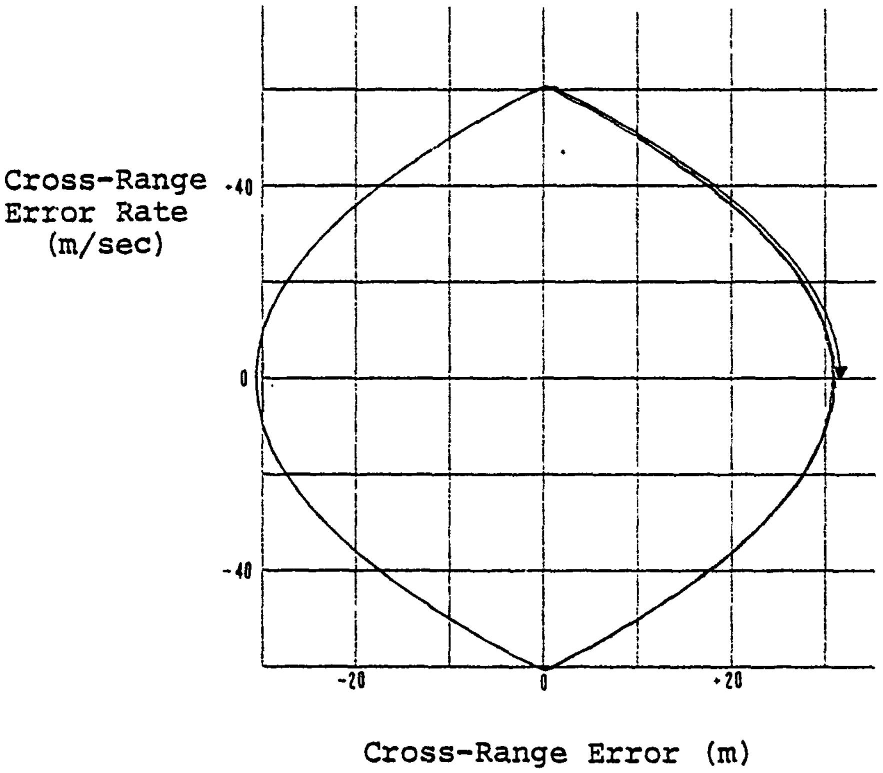


Figure V-2 Phase Plane When Control Function By-Passed With Initial Values Introduced for Cross-Range Error and Error Rate

just after missile launch, this "basic" guidance system does not dampen their effects. This increases the probability of having a greater miss distance. With an initial cross-range error of ten meters and with an initial cross-range error rate of fifty meters per second applied, the undamped effect, plus increased cross-range error with time, combine to cause the missile to miss the target. Results are shown in Figures V-2 through V-4. Although not readily apparent in Figure V-4, the

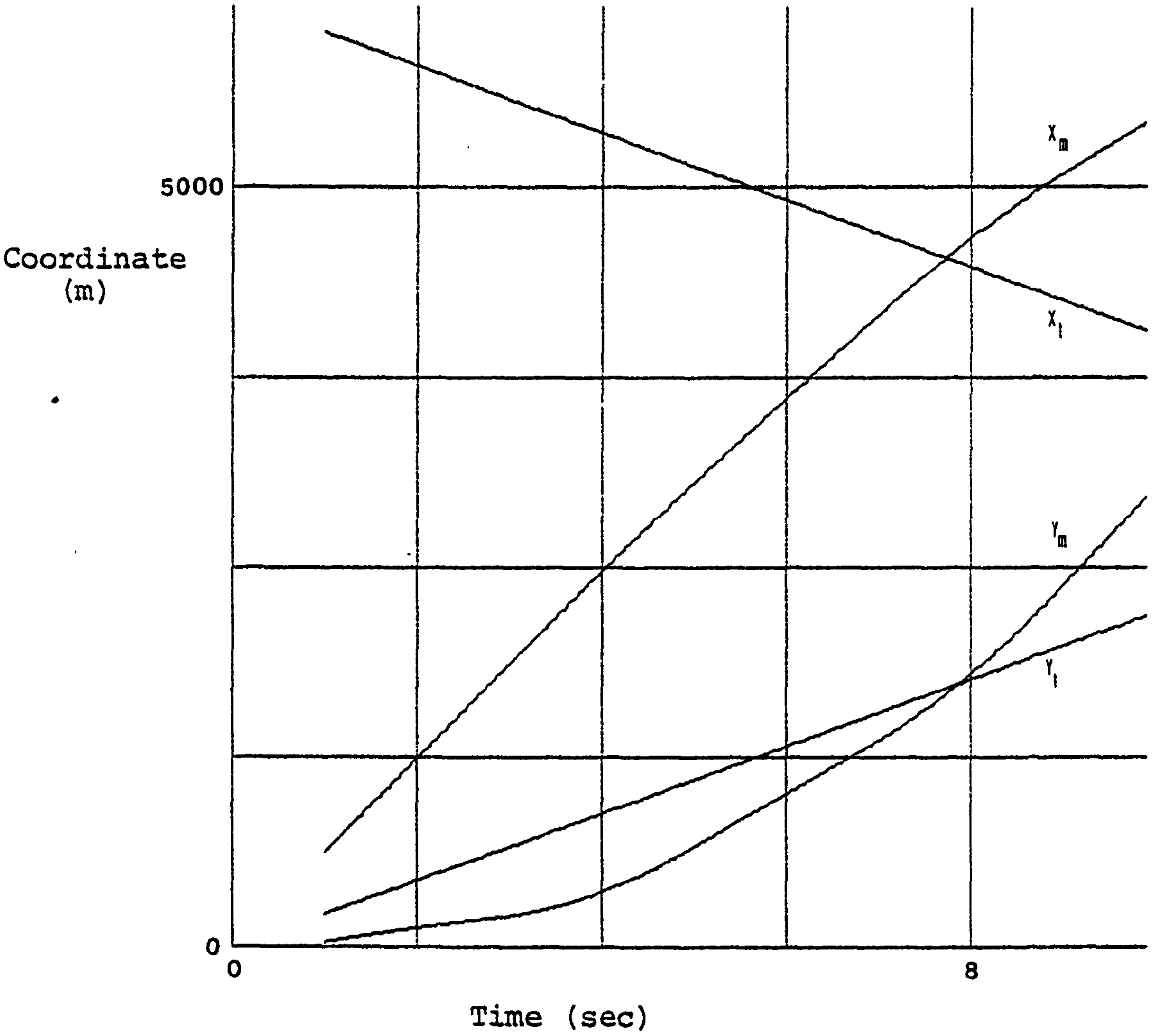


Figure V-3 Missile & Target Positional Coordinates When Control Function By-Passed With Initial Values Introduced for Cross-Range Error and Error Rate

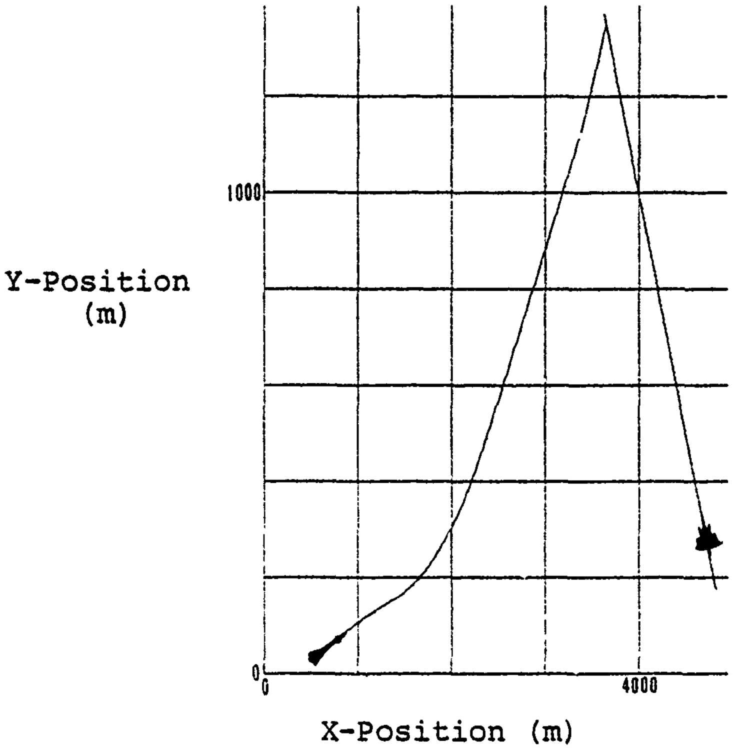


Figure V-4 Missile & Target Flight Paths When Control Function By-Passed With Initial Values Introduced for Cross-Range Error and Error Rate

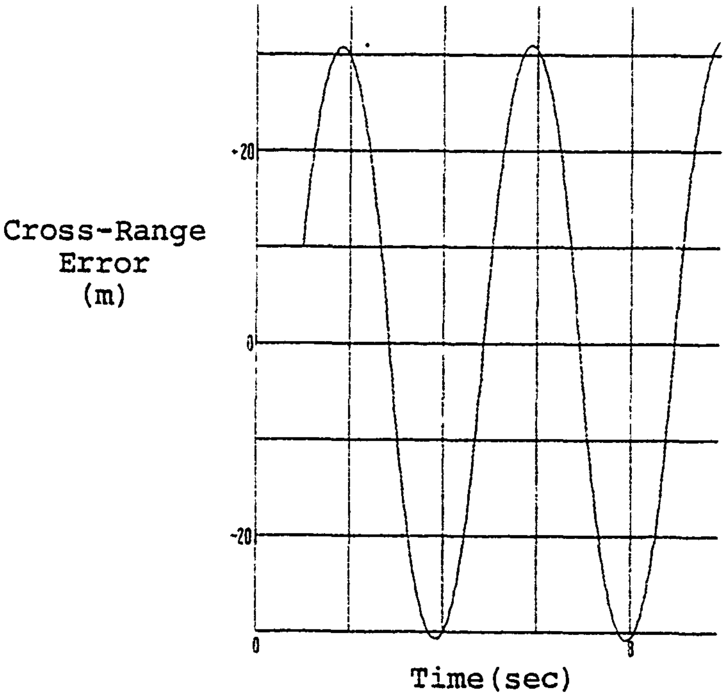


Figure V-5 Missile Cross-Range Error When Control Function By-Passed With Initial Values Introduced for Cross-Range Error and Error Rate

missile moves about the line of sight with a cross-range error varying as shown in Figure V-5. The cross-range error does increase by a small amount with time as shown in Figure V-1.

The models discussed later are somewhat unrealistic because they assume that the required force obtained from the missile's thrust-vector control can be applied instantaneously at the proper time and reversed in a similar fashion to decelerate the missile. This, however, does provide optimum missile performance for comparison with any real system.

In all the simulations the missile is acceleration-limited with instantaneous application of constant side-force assumed. One could allow for a time delay in switching the missile's thrust-vector control from the positive to the negative direction. However, this would raise the order of the analysis. The computer program, though, can be modified to handle systems to the thirtieth order.

Optimum acceleration switching requires that a switching-boundary be established that zeros the error in the minimum time. That boundary was determined to be parabolic as drawn in the phase-plane representation of Figure V-6 and as supported

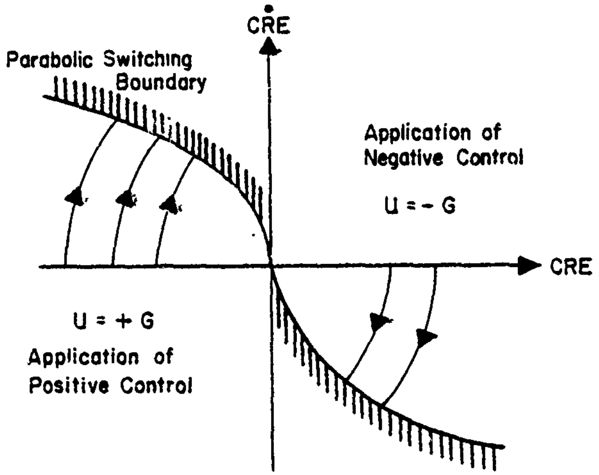

$\dot{\text{CRE}}$
Parabolic Switching Boundary
Application of Negative Control
U = - G
CRE
U = + G
Application of Positive Control

Figure V-6 Optimum Parabolic Switching Boundary

by the calculations which follow later. When the acceleration or deceleration trajectories attempt to cross this optimum curve (switching boundary) the system switches to that optimum path and the error is driven to zero along it.

In order to verify this result, a phase plane was produced by introducing, as initial conditions, large cross-range error and cross-range error rates into a computer-simulated guidance model containing acceleration-limiting methods within the tracker control function. The resulting phase plane, Figure V-7, substantiates that the tracker control function switches properly to the theoretically-optimum, parabolic curve for error reduction expected by calculation. This curve is represented in Figure V-6. Also, the target was intercepted by the missile as substantiated in Figures V-8 and V-9. Figure V-10 shows the

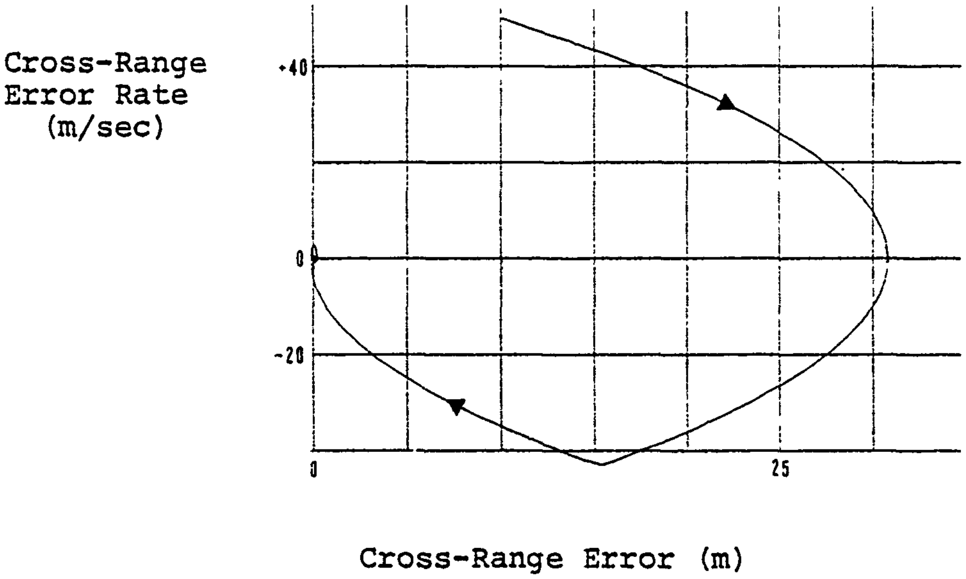


Figure V-7 Phase Plane When Control Function Utilized With Initial Errors Introduced for Cross-Range Error and Error Rate

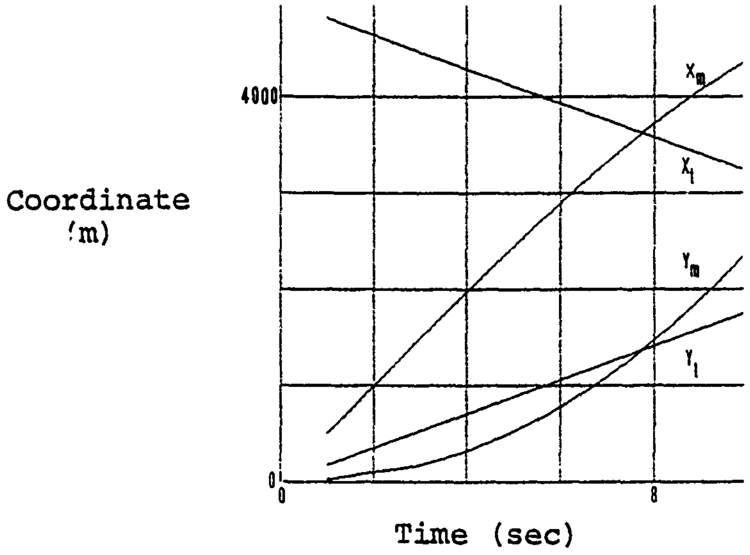


Figure V-8 Missile & Target Positional Coordinates When Control Function Utilized With Initial Errors Introduced for Cross-Range Error and Error Rate

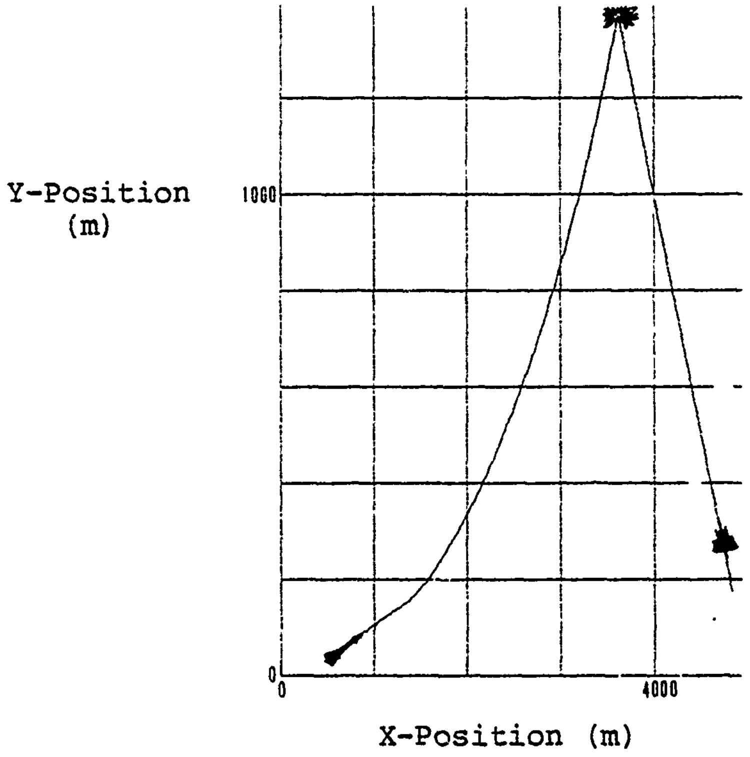


Figure V-9 Missile & Target Flight Paths When Control Function Utilized With Initial Errors Introduced for Cross-Range Error and Error Rate

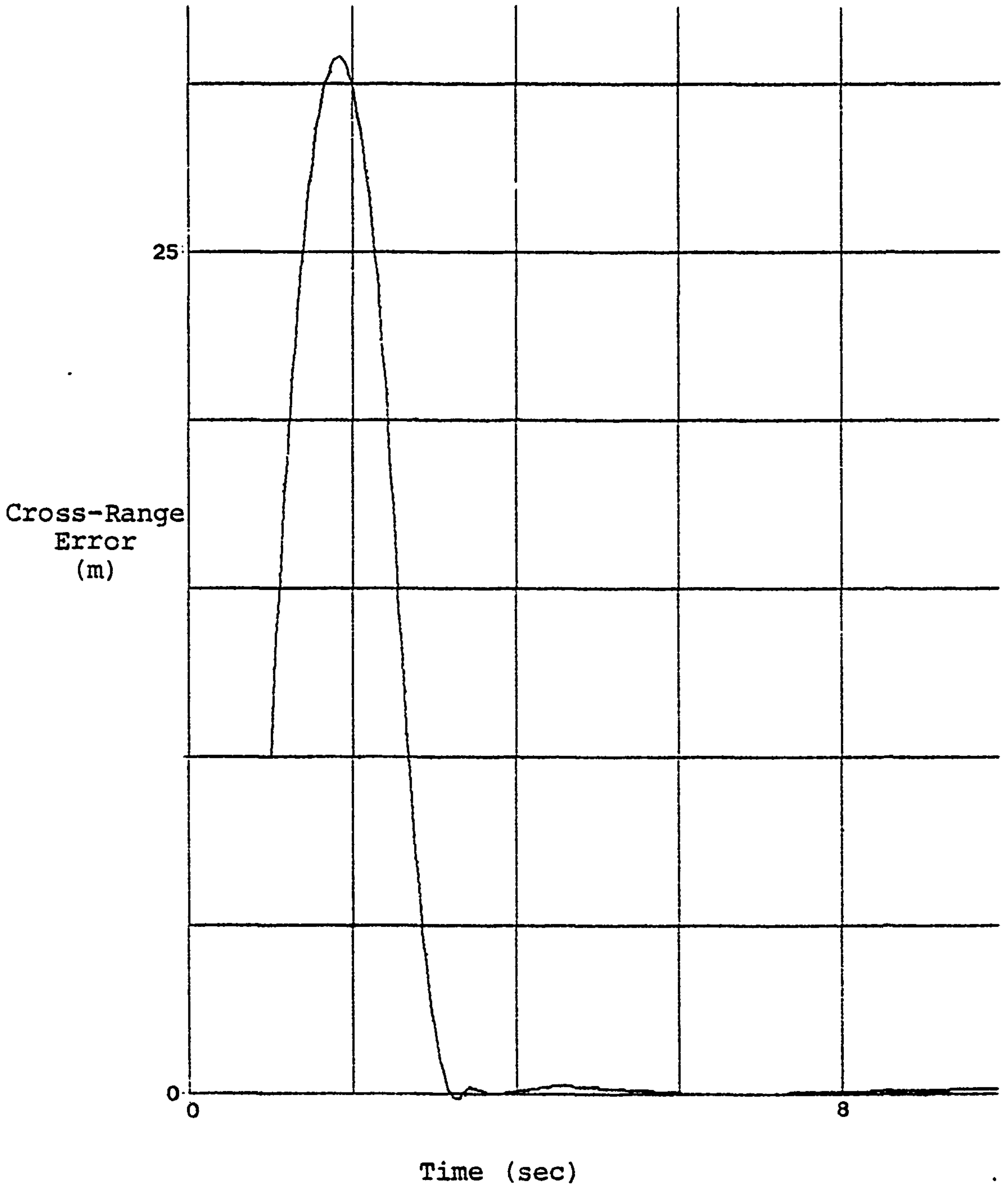


Figure V-10 Missile Cross-Range Error When Control Function Utilized With Initial Errors Introduced for Cross-Range Error and Error Rate

missile's cross-range error with time of flight. This error is not readily apparent in Figure V-9 because of the relatively large distance scale utilized in that figure. As can be seen this cross-range error is reduced quickly.

The theoretical calculations supporting the parabolic nature of the switching boundary represented by Figure V-6 follow:

$m$ = missile mass

$F$ = lateral force applied to missile

$CRE$ = cross-range error

$\dot{CRE}$ = cross-range error rate

$\ddot{CRE}$ = cross-range error acceleration

$G$ = magnitude of cross-range missile acceleration

$U$ = cross-range missile acceleration

$C$ = constant

Since the mass of the missile and the magnitude of the output force from the missile thrust-vector control are assumed constant, the magnitude of the acceleration is constant, thus:

$$m\,\ddot{CRE} = |F|$$

$$|\ddot{CRE}| = |F|/m = G$$

$$\left|\frac{d}{dt}\dot{CRE}\right| = G$$

$$\int |d\,\dot{CRE}| = \int G\,dt \rightarrow |\dot{CRE}| = Gt + C$$

Let the initial conditions for time and cross-range error be zero. Therefore, the constant, "C", is zero also.

$$\dot{\mathrm{CRE}} = \mathrm{G}t \;,\; \dot{\mathrm{CRE}} \geq 0$$

$$t = \dot{\mathrm{CRE}}/\mathrm{G} \;\rightarrow t^2 = \dot{\mathrm{CRE}}^2/\mathrm{G}^2$$

$$\frac{d}{dt}\,\mathrm{CRE} = \mathrm{G}t \qquad \int d(\mathrm{CRE}) = \int \mathrm{G}t\;dt$$

$$\mathrm{CRE} = \mathrm{G}t^2/2 + |\mathrm{CRE}(0)|,\; t^2 = \dot{\mathrm{CRE}}^2/\mathrm{G}^2$$

$$\mathrm{CRE} = \dot{\mathrm{CRE}}^2/2\mathrm{G} + |\mathrm{CRE}(0)|,\quad \mathrm{CRE} \geq 0$$

Since the direction of the missile's thrust-vector control force can be either in the positive or negative direction, the sign of the cross-range error (CRE) must be preserved. The initial missile cross-range error can be in either direction; consequently, the equation becomes:

$$\mathrm{CRE} = \dot{\mathrm{CRE}} \quad |\dot{\mathrm{CRE}}|/2\mathrm{G} + \mathrm{CRE}(0)$$

This equation is parabolic as depicted in Figure V-6.

Also, since the magnitude of the missile cross-range acceleration, "G", is constant, the corresponding acceleration vector, "U", is:

$$\mathrm{U} = \pm\mathrm{G} \;\rightleftarrows\; \mathrm{U} = (-\mathrm{G})\mathrm{SIGN}(\text{error function})$$

$$,\ \text{error function} = \mathrm{EX} = \dot{\mathrm{CRE}}|\dot{\mathrm{CRE}}|/2\mathrm{G} + \mathrm{CRE}$$

This drives the missile's control surfaces in the proper direction, a direction which is opposite to the sign of the error function, thus moving the missile back toward the line of sight with a constant magnitude of acceleration, "G".

A simplified block diagram representing the command to line-of-sight guidance system is presented in Figure V-11.

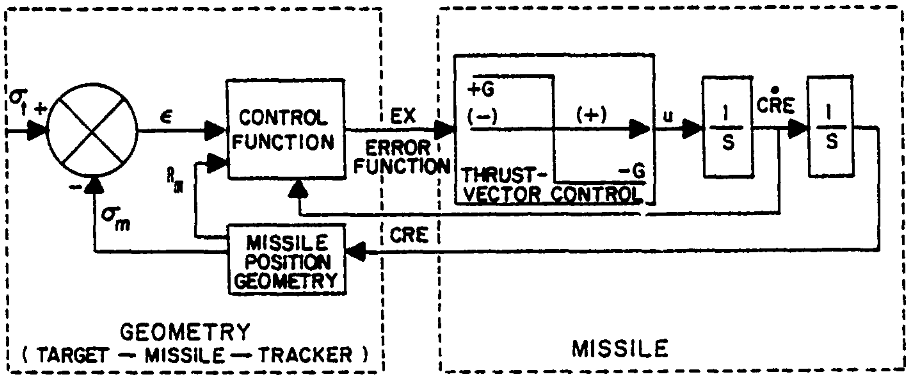


Figure V-11 Elementary Command to Line-of-Sight Block Diagram

# VI. 'BASIC' COMMAND TO LINE-OF-SIGHT SIMULATION

## A. PROGRAM DESCRIPTION

Utilizing the concepts previously discussed, this program simulates theoretically optimum surface-to-air missile maneuver against a constant-velocity target in two dimensions. The engagement was designed to occur within the first quadrant of a standard Cartesian-coordinate reference system with the ground tracker, missile lauching unit, located at the origin. The missile was assumed to be "captured" by the guidance system approximately one second after missile launch or at a distance of about 500 meters from the ground tracker unit. At the time of missile "capture" an initial cross-range error of ten meters and a cross-range error rate of 50 meters per second were introduced to demonstrate the ability of the guidance system to neutralize initial errors. It should be noted that these values of error were selected entirely arbitrarily and that they could be any reasonable values assumed to exist at the time of missile "capture".

The target was "flown" across the first quadrant from a position 5000 meters from the tracker on the X-axis (5000,0) toward a position 5000 meters from the tracker on the Y-axis (0,5000).

In order to use a command to line-of-sight guidance scheme effectively, the missile must possess a speed advantage over the target. Therefore, in this model the missile velocity was chosen to be 500 meters per second versus a 250 meter-per-second target. At sea level these speeds equate to approximately

Mach 1.5 and Mach 0.75 respectively. The missile, also, after being launched directly toward the target, was maneuvered using on-off, thrust-vector control which applied a lateral acceleration of 60 meters per second squared to the missile. This is just greater than six times the earth's gravitational acceleration at sea level.

In general, keeping in mind the scenario above, the program operates as follows:

> The equations for the target's movement were written and appropriately placed into the program so that the target and missile angular directions could be compared. The difference between these two angles creates an error angle. This error angle is geometrically converted by the tracker computer into the cross-range error, the distance of the missile from its intended flight path - the line of sight to the target in this case. That error distance is reduced to zero by applying a lateral, thrust-vector-controlled acceleration to the missile in the correct direction as determined and commanded by the ground tracker unit. The optimum parabolic switching boundary, discussed in Section V, was used in the tracker computations. In order to determine this cross-range error, the tracker estimates the missile's range by the elapsed time of flight and the missile's velocity profile. Also, to simplify the program somewhat, the

missile's velocity was assumed to be constant in a radial direction from the tracker. When the cross-range error is reduced to an acceptable distance, the amount of thrust-vector control is limited to prevent rapid changes between full thrust-vector deflections in both directions. If the error, again, becomes unacceptably large, the full deflection positions are utilized. Missile position is determined by state variable analysis using the INTEG2 subroutine for integration of the lateral thrust-vector-controlled missile acceleration. The use of this subroutine is explained in Appendix A. After two integrations, the missile's new cross-range error is determined and, then, this distance is used with estimated missile range to provide a revised missile angular position. This process is repeated in increments of five thousandth of a second from the time of missile "capture" by the tracker, a time of approximately one second, until maximum missile range is attained after ten seconds of flight. Thus, the missile is driven continually to a position upon the line of sight to the target The program variables are "frozen" when the missile reaches the closest point of approach to the target if the value of constant C(1) is one. If this value is zero, the program computes until the time of missile flight reaches ten seconds.

The program was written in the FORTRAN IV language. A detailed, annotated copy of the program, including a table of program variable definitions, is provided following the narrative portion of this paper.

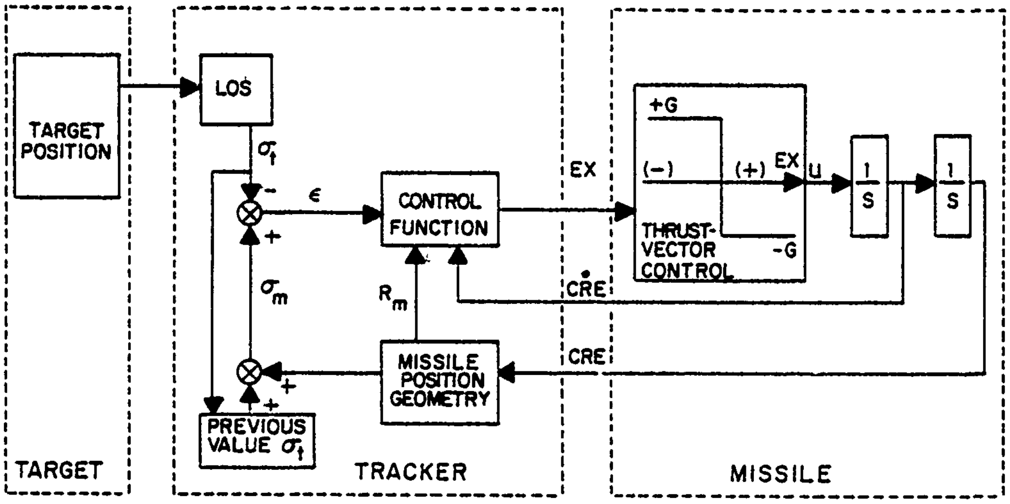


Figure VI-1 'Basic' Computer Simulation Block Diagram

B. SYSTEM BLOCK DIAGRAM

The command to line-of-sight computer simulation block diagram, Figure VI-1, shows how the elementary block diagram of this guidance technique, Figure V-11, was expanded upon in order to develop a computer model for the system. As will be seen, while appearing to be quite simple in concept, the simulation becomes increasingly more complicated as its sophistication is enhanced.

C. SIMULATION RESULTS

The plotted output of the phase plane, Figure VI-2, shows how the cross-range error and cross-range error rate are driven to a near zero condition along the optimum, parabolic, switching boundary. Once the cross-range error is less than one meter, the magnitude of the thrust vector control is reduced by a sixth to minimize the possibility of over-controlling the missile.

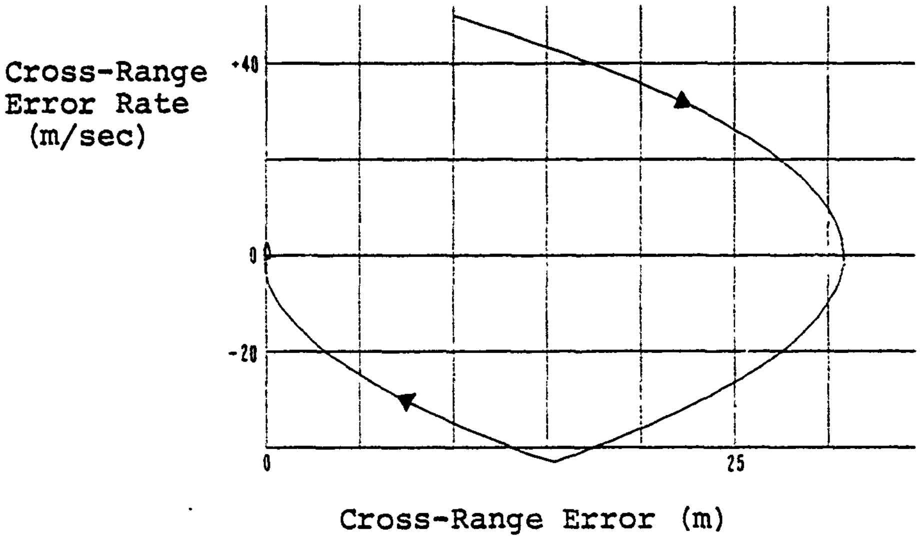


Figure VI-2 'Basic' Simulation Phase Plane

Figure VI-3 presents the "X" and "Y" positions for the missile and the target versus time. As can be seen the intercept occurs in just under eight seconds after missile launch.

Figure VI-4 indicates the relative positions of the missile and target with the terminal points for each flight profile stopped at the closest point of approach. An intercept did occur.

In order to ensure that the missile's cross-range error from the line of sight was damped, Figure VI-5 was produced to show this cross-range error versus time of flight. The initial errors, applied at the time of missile "capture" by the

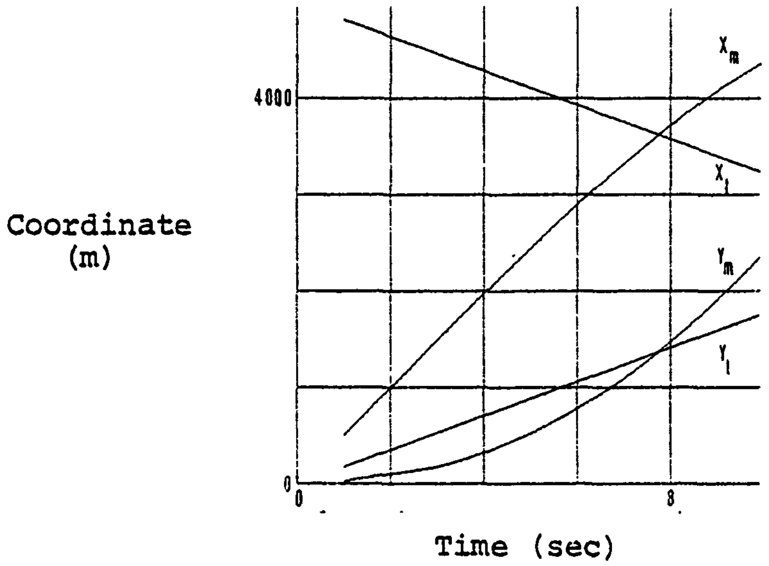


Figure VI-3 'Basic' Simulation Missile & Target Positional Coordinates

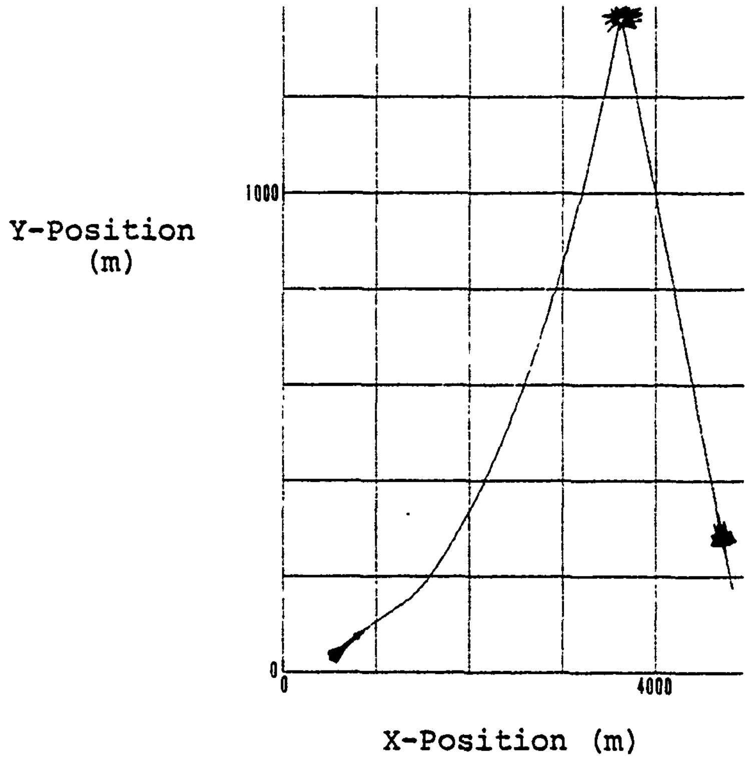


Figure VI-4 'Basic' Simulation Missile & Target Flight Paths

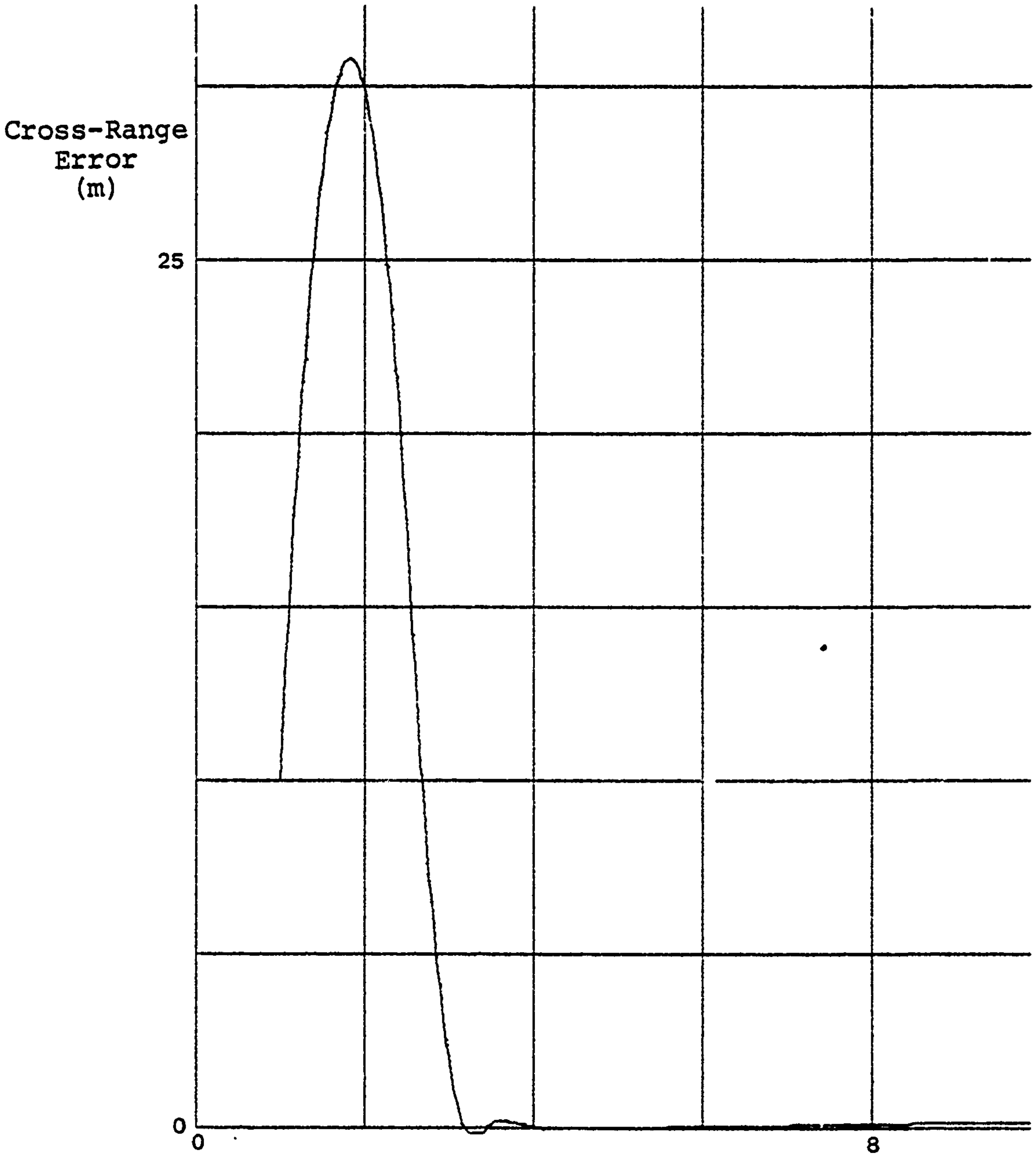


Figure VI-5 'Basic' Simulation Missile Cross-Range Error guidance system (corresponding to a time of approximately one second in the figure), were reduced sufficiently in less than two seconds.

## D. EFFECTS OF DECEPTIVE JAMMING

A readily apparent deceptive jamming technique, which might be effective against this class of guidance system, would be to employ, aboard the target aircraft, an emitter which imitates the signal that is used by the tracker to locate the missile's angle from the reference plane. This could be, for example, depending upon how the missile is tracked in angle, an imitation of the missile's transponder radio-frequency signal or a duplication of the missile's infrared emissions. In any case the deceptive jamming signal from the target aircraft should be of sufficient strength to ensure that the missile tracking system within the ground-located tracker considers the target to be the missile. There are, most assuredly, counter-countermeasures which could be applied to negate this deceptive technique; however, this, currently, would be one method of generating a zero error signal between the missile and target angles. Additionally, this deceptive countermeasure provides a method to demonstrate how the computer simulation drives the missile with inputs of erroneous guidance information.

In order to implement this deceptive jamming technique, it was assumed that the target aircraft jamming equipment became effective when the missile was approximately half the distance to the target. This would equate to about four seconds of missile flight time. Therefore, since this jamming countermeasure system simply zeros the error angle, only one additional equation was necessary to modify the previously discussed "basic" command to line-of-sight guidance model. The equation, IF(T.GE.4.0)E=0., was placed into the program following the computational step for the calculation of the angular error, E=SIGM-SIGT.

The resulting missile trajectory and other parameters are readily apparent in Figure VI-6 through VI-8. As shown in Figure VI-6, the error was reduced to zero by the time that jamming commenced. After deceptive jamming was initiated, the cross-range error rapidly increased causing a miss of over 300 meters. Needless to say, because of the short times of flight involved, the countermeasures must be employed in a timely manner. Otherwise, the missile could approach, easily, within lethal range of the target aircraft.

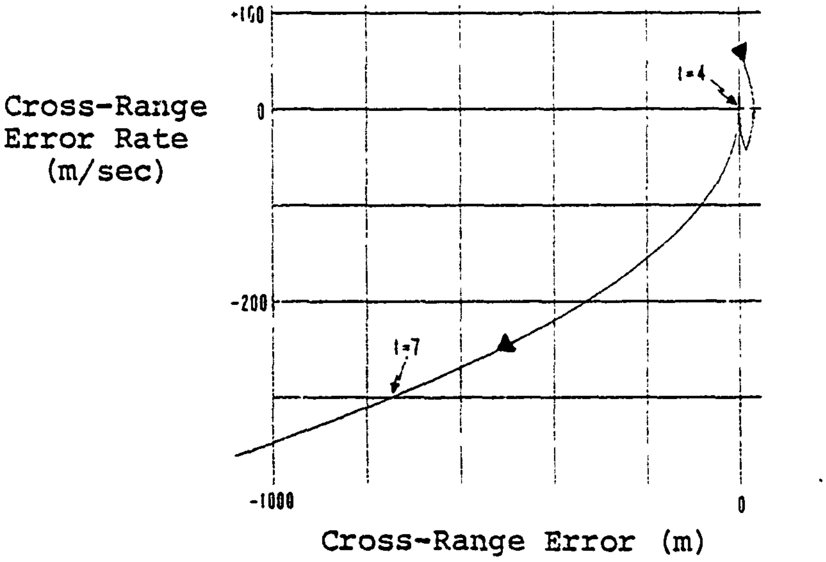


Figure VI-6 'Basic' Simulation Phase Plane When System Deceptively Jammed

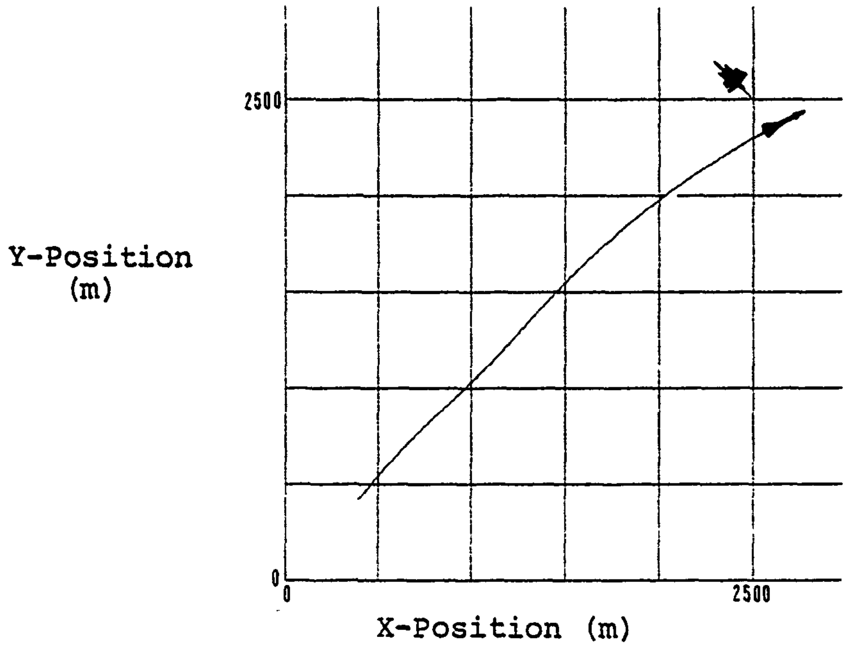


Figure VI-7 'Basic' Simulation Missile & Target Flight Paths When System Deceptively Jammed

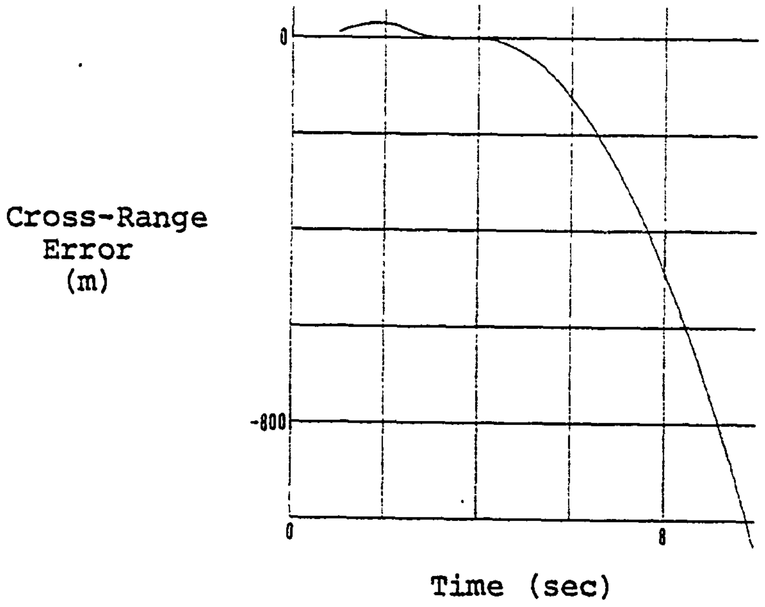


Figure VI-8 'Basic' Simulation Missile Cross-Range Error When System Deceptively Jammed

# VII. 'LEAD-ANGLE' COMMAND TO LINE-OF-SIGHT SIMULATION

## A. PROGRAM DESCRIPTION

This 'lead-angle' simulation uses the same target profile as that of the 'basic' command to line-of-sight simulation. This allows a comparison of the two simulations to a certain degree. One parameter, which can not be compared accurately, is the time to impact. This occurs because the radial missile velocity from the tracker was assumed to be a constant value - in this case 500 meters per second. Therefore, intercepts will take place at about the same time regardless of missile maneuvers. A general comparison can be made, however, by mentally comparing the plotted missile flight profiles, the longer profile taking the most time in an actual engagement. Also, in order to avoid redundancy, discussion will be limited to modifications of the previously described computer model. Only a few changes were required to produce a 'lead-angle' variant for the command to line-of-sight guidance system.

In this guidance scheme, the spacial positions of both the target and the missile must be accurately and continuously known in order to compute an advanced impact position. This is accomplished by calculating the distance from the missile to the target and their closing velocity. From this data, a time remaining until impact can be estimated. This remaining time of flight, then, is combined with the target's current velocity vector and position to ascertain the expected impact point. A synthetic line of sight toward this location is generated by the ground tracker unit computer and the missile is commanded to "fly" up this flight path instead of along the line of sight directly toward the target. The guidance scheme used to maintain the missile on course, the synthetic line of sight, is, otherwise, identical to that of the "basic" simulation previously described. Once the missile passes the closest point of approach to the target, this simulation ceases to be valid. This is quite all right because, once the missile has passed the target, no maneuver would be sufficient to effectively engage the aircraft.

## B. SYSTEM BLOCK DIAGRAM

The "lead-angle" command to line of sight computer simulation block diagram is contained in Figure VII-1. It is essentially a modification of the "basic" simulation block diagram, Figure VI-1. As described previously, target and missile positions plus their velocities must be known for the guidance scheme to be viable.

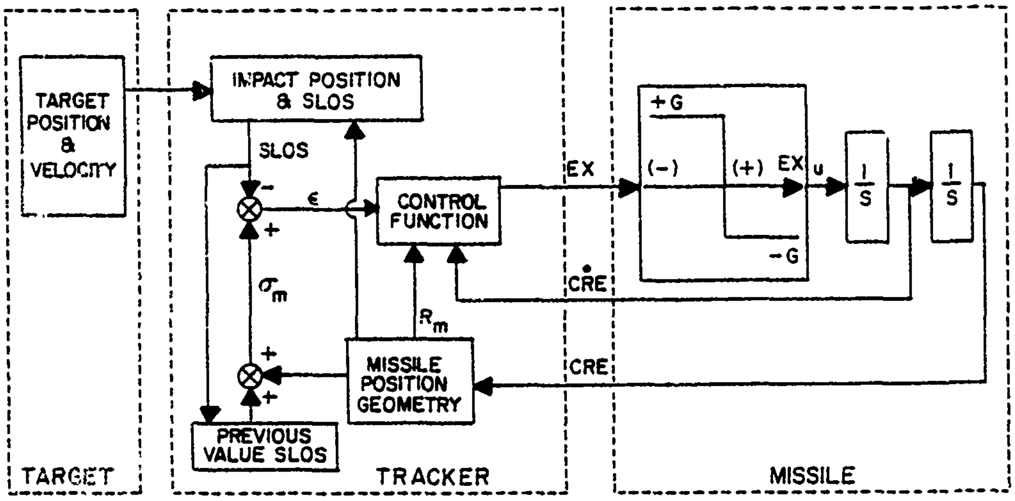


Figure VII-1 'Lead-Angle' Computer Simulation Block Diagram

## C. SIMULATION RESULTS

This simulation supports the idea that the use of a "lead-angle" variant to the "basic" command to line-of-sight guidance system enhances the effectiveness of the elementary guidance technique. As can be seen in Figures VII-2 through VII-5, the missile was "captured" by the guidance system at a time of approximately one second from launch. The initial cross-range and cross-range-rate errors were quickly reduced and the missile "flew" out the synthetic line-of-sight to successfully intercept the target aircraft. Of course, if the target employs electronic countermeasures such as denial jamming of the target tracking equipment, this "lead-angle" method would prove to be useless because accurate positioning information must be maintained to compute the estimated impact point. Also, if the target maneuvers sufficiently to cause the impact point to vary at high rates, the synthetic line of sight may shift at too high

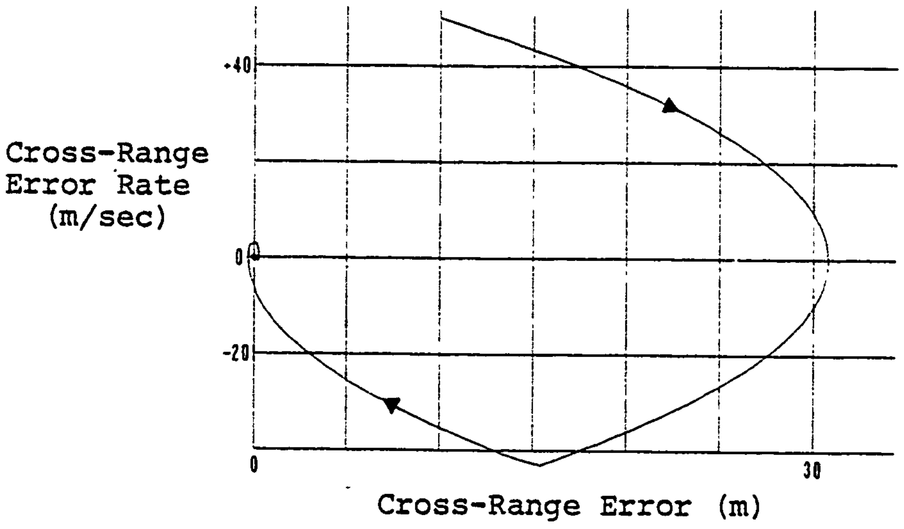


Figure VII-2 'Lead-Angle' Simulation Phase Plane

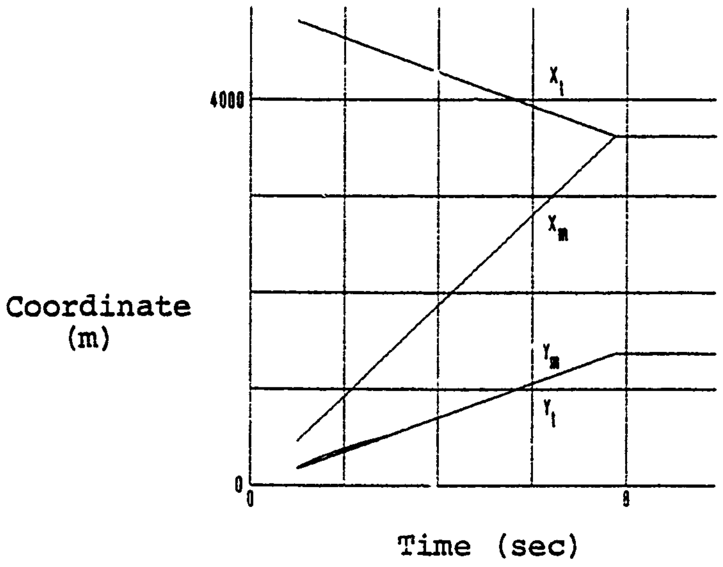


Figure VII-3 'Lead-Angle' Simulation Missile & Target Positional Coordinates

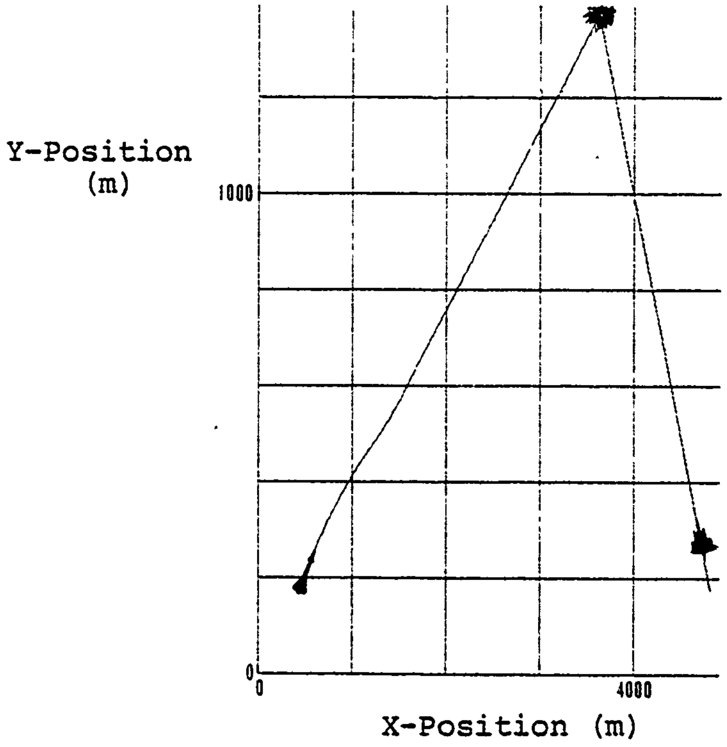


Figure VII-4 'Lead-Angle' Simulation Missile & Target Flight Paths a rate for the missile to follow. Thus, the missile may not be able to physically engage the impact point but might be able to still effectively attack the target by the "basic" guidance scheme. Consequently, in order to capitalize upon the advantage of the "lead-angle" variant (an increased effective missile envelope against a relatively slow maneuvering target which is not employing electronic countermeasures), it seemed that a combined guidance technique should be considered. Therefore, the "consolidated" simulation discussed in the next section was contrived.

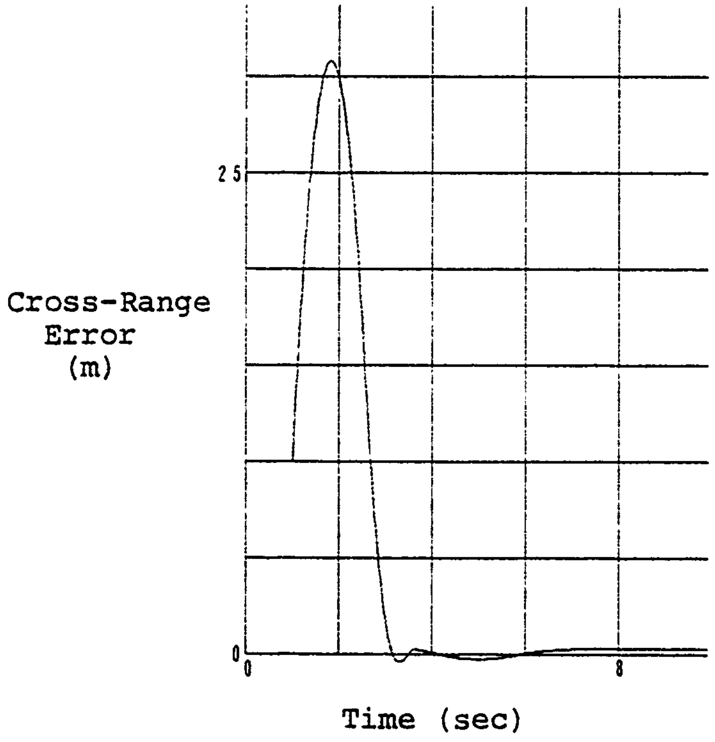


Figure VII-5 'Lead-Angle' Simulation Missile Cross-Range Error

# VIII. 'CONSOLIDATED' COMMAND TO LINE-OF-SIGHT SIMULATION

## A. PROGRAM DESCRIPTION

When the two previous simulations were combined, a major problem surfaced. That problem was basing the missile's new position upon two different references, the line of sight to the target in the case of the "basic" model and the synthetic line of sight to the impact point in the case of the "lead-angle" variant. This incompatibility caused the missile's position to jump radically as the "consolidated" simulation attempted to shift alternately between the two reference lines as the jamming situation varied. Therefore, a single reference had to be implemented for continuous use regardless of whether jamming was or was not encountered. The X-axis of the Cartesian-coordinate system was selected since various angular measurements were based upon it already. Also, this has proved to provide a more accurate model of the missile's optimum flight profile because the reference does not move during the time of missile flight. If a strictly "basic" simulation should be desired, jamming is considered to be effective during the entire engagement. Conversely, the absence of jamming during the missile's flight produces a total "lead-angle" intercept of the target aircraft. Of course, different duty cycles for jamming can be implemented by the target during a single computer run to test the ability of the missile system to transition between the two attack modes. A limited number of different jamming combinations were tested and are discussed later.

In order to implement jamming, the jamming time increments are read into the simulation through five time intervals specified on the first five data cards. These cards preceed those required by the INTEG2 subroutine as explained in Appendix A. The time each jamming interval begins is punched into columns 1-10 of each card in F-format. The times that jamming ceases are entered in columns 11-20. Should a no-jamming condition be desired, then, all five cards are left blank but they must not be ommitted. Total jamming can be obtained by making one card cover the entire time of flight. For example, the first card could have columns 1-10 blank and the value 11.0 entered into columns 11-20. This produces jamming from time zero until a time of eleven seconds, which is well after the end of the engagement. The remaining four cards are blank and must be inserted.

This simulation "drives" the missile in the "Y" direction with sufficient "Y" velocity to reduce the cross-range error to zero. The missile's X-direction velocity component is used to maintain a constant, outward, radial velocity, "$V_m$", toward the target. In this simulation, as in its predecessors, that radial velocity, "$V_m$", is assumed to be 500 meters per second. Therefore, the velocity of the missile in the "X" direction is related to that obtained from the forcing function in the "Y" direction. This is shown in Figure VIII-1.

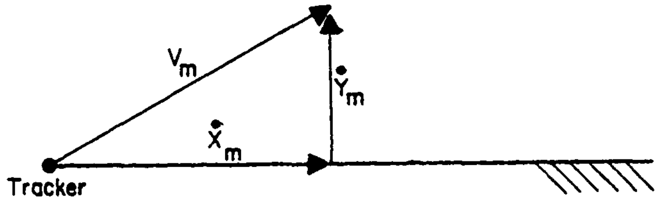
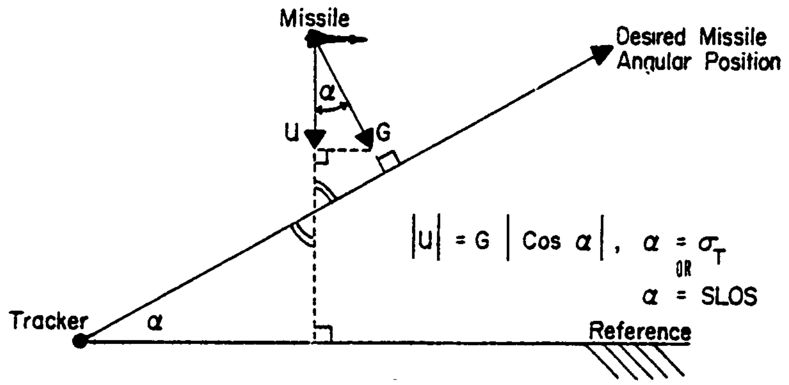


Figure VIII-1 'Consolidated' Simulation Missile Velocity Vectors

The missile's Y-direction velocity is obtained from the state variable analysis through a single integration of the forcing function, "U", which is the missile's Y-direction acceleration. This acceleration is the "Y" component of the cross-range error acceleration achieved by thrust-vector control of the missile as shown in Figure VIII-2.



Missile
Desired Missile Angular Position
α
U
G
$|U| = G\ |\text{Cos}\ \alpha|,\ \alpha = \sigma_T$ OR $\alpha = \text{SLOS}$
Tracker
α
Reference

Figure VIII-2 'Consolidated' Simulation Missile Acceleration Vectors

The missile's Y-coordinate position, then, is the second integration of the forcing function. From these results, the X-direction components were determined since the velocities must yield a constant, radial velocity of 500 meters per second. In order to ensure a correct integration process, the missile's "Y" velocity was not limited to 500 meters per second; however, it never exceeded that value in any of the computer runs tested. It should be noted that the program will zero the acceleration and the velocity of the missile in the "X" direction if excessive missile "Y" velocity is experienced. Also, the results, then, will become invalid; consequently, the velocity of the missile,

"$V_m$", should be increased or the target's speed reduced to obtain a theoretically-possible engagement scenario.

The targets, in the previous simulations, "flew" a constant-velocity, linear course. In this simulation the target was maneuvered sinusoidally in the "Y" direction for the purpose of observing the effects of a moving impact point. The maximum velocity of the target in the "Y" direction is 64 meters per second with a half period of four seconds. The target's "X" velocity was a constant, negative 210 meters per second. This did produce some interesting results. First, this type of target maneuver caused the impact position to move at even higher rates than the target moved. Thus, in some cases, the missile may not be able to engage, adequately, the impact point which is moving at a very high velocity. It, however, may be able to attack the target itself by utilizing the "basic" command to line-of-sight approach. To determine which situation exists necessitated the use of the impact point's velocity to compute the synthetic line of sight, the angle upon which the missile "flies" in a non-jamming environment. If this impact-point velocity exceeded eight-tenths of the missile's velocity, then, the missile was commanded to "fly" directly toward the target's present position by the "basic" guidance method. This was in lieu of utilizing the "lead-angle" approach with only an unstable impact position and, thus, an erratic or too quickly moving synthetic line of sight.

In order to provide a longer time of flight, the target's initial position was moved to coordinates (6000,1000) and the target was "flown" inbound at 210 meters per second while sinusoidally maneuvering about the 1000 meter Y-ordinate. Also, in regard to the short time of flight involved, excessively large initial errors in either distance or rate should be avoided because of the time required to neutralize these errors. For example, since the missile's maximum acceleration is only 150 meters per second squared, initial erroneous conditions of 250 meters and 250 meters per second for the missile's Y-direction components require many seconds to correct. However, with initial conditions of 150 meters and 150 meters per second, the simulation proved the missile to be effective against this target over a full scope of jamming duty cycle variations - from none to continuous denial jamming.

The initial angle to the missile at the time of "capture" by the guidance system, a time of one second or 500 meters from the launcher-tracking unit, is determined by the initial conditions, X(1) and X(2). As can be recalled, X(1) is the missile's "Y" position at the time that the weapon is 500 meters radially from the launcher. Therefore, the angle is the inverse sine of X(1) divided by 500 meters. In order to attain this "Y" position in one second, the missile's "Y" velocity, then, was approximately the "Y" distance per second. Consequently, in this simulation, initial conditions of 100 meters for X(1) and 100 meters per second for X(2) equate to an angle of 11.54 degrees above reference.

Also, the magnitude of the missile's maximum lateral acceleration, "G", was increased to 150 meters per second squared. This provided a sufficient forcing function to allow engagement of the target. This was possible even with increased initial errors at the time of missile "capture" by the guidance system; with a maneuvering target accelerating at just over 50 meters per second squared, at maximum, in the "Y" direction; and with various jamming duty cycles from zero to one. As in the previous simulations, the thrust-vector control of the missile was reduced by one-sixth; however, this was done only when the sum of the absolute values of cross-range error and error rate was less than or equal to four vice the value of one, which was used previously. The error tolerance was increased to provide a larger "idling" area for the missile's thrust-vector control because of the increased forcing function utilized.

The actual computer program for the simulation, with comments included, follows the appendices.

## B. SYSTEM BLOCK DIAGRAM

The block diagram of the "consolidated" simulation is contained in Figure VIII-3. Although it appears formidable at first glance, the diagram merely combines the concepts of the two previous simulations with slight modifications to account for the use of a new reference from which to compute the missile's location. If the tracking unit ascertains that it is contending with either denial jamming or a highly-maneuvering target, then, only the upper portion of the tracker diagram is used for guidance. Conversely, if both a non-jamming condition and a relatively slowly-maneuvering target situation are experienced, the lower portion of the tracker diagram produces guidance for the missile.

The simulation does produce necessary calculations in all areas of the tracker diagram continually throughout the engagement, but only generates commands to the missile from the appropriate section of the model.

After the program reads the data cards, the times of jamming are known and the appropriate control function within the tracker's computer is implemented to control the missile. The magnitude of the impact point's velocity is considered, also, to ascertain the stability of the synthetic line of sight and, thus, to determine whether target maneuvers make the "lead-angle" approach of guidance inappropriate.

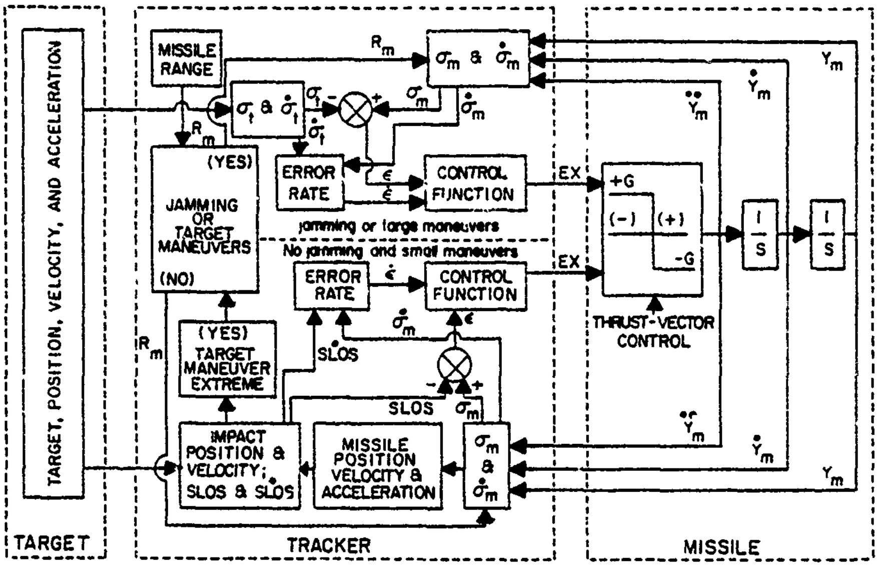


Figure VIII-3 'Consolidated' Computer Simulation Block Diagram

## C. SIMULATION RESULTS

Three engagements were computed under different jamming conditions. They were continuous jamming, intermittent jamming having an approximate duty cycle of one-half, and no jamming. During intermittent jamming, the ground-located tracker unit was denied range information within the following time increments after missile launch: 1.0-3.0, 4.0-5.0, 6.6-7.0, 7.4-7.8, and 8.2-8.6 seconds.

Utilizing the scenario which was previously discussed, all three attacks resulted in successful intercepts of the target. As can be observed in Figure VIII-4, the best flight profile of the three simulated runs occurred when total jamming was experienced. This jamming caused the missile to "fly" directly toward the target.

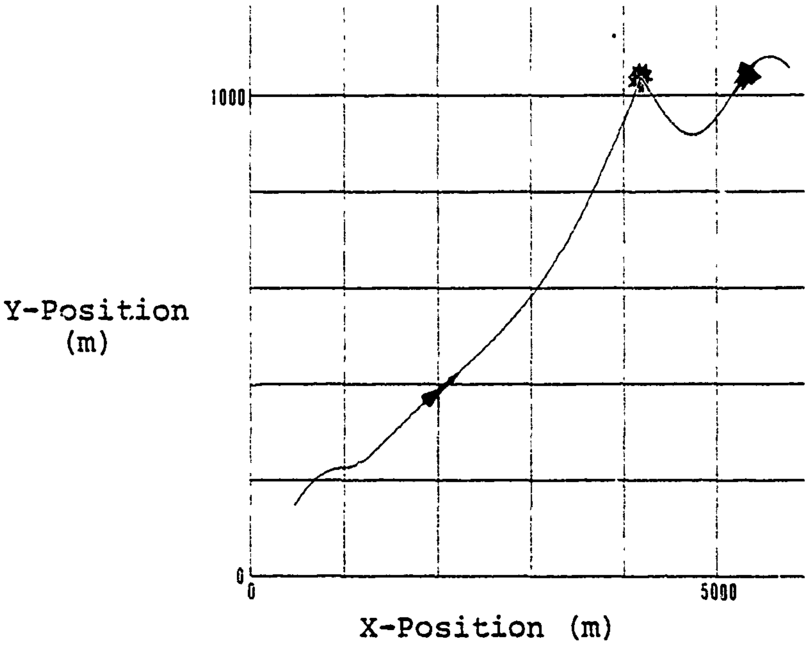


Figure VIII-4 'Consolidated' Simulation Missile & Target Flight Paths When Continuously Jammed

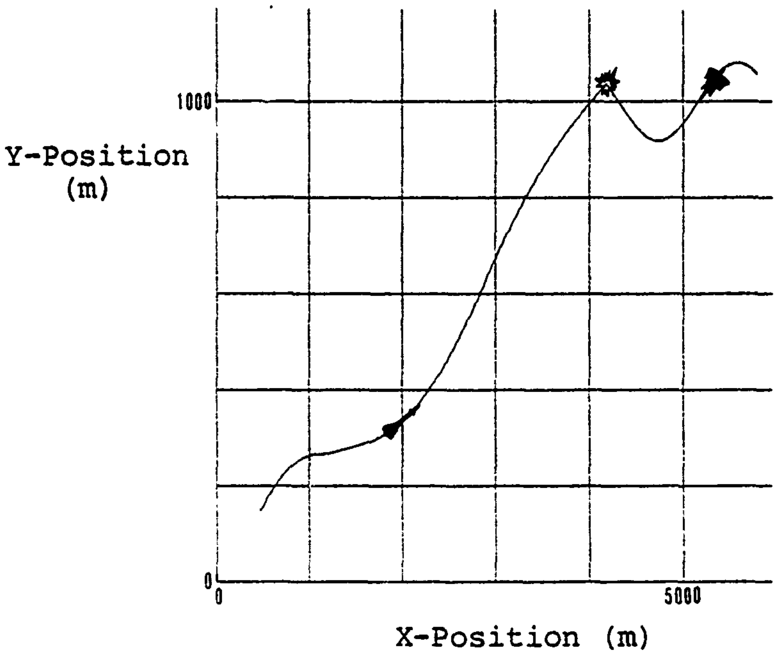


Figure VIII-5 'Consolidated' Simulation Missile & Target Flight Paths When Not Jammed

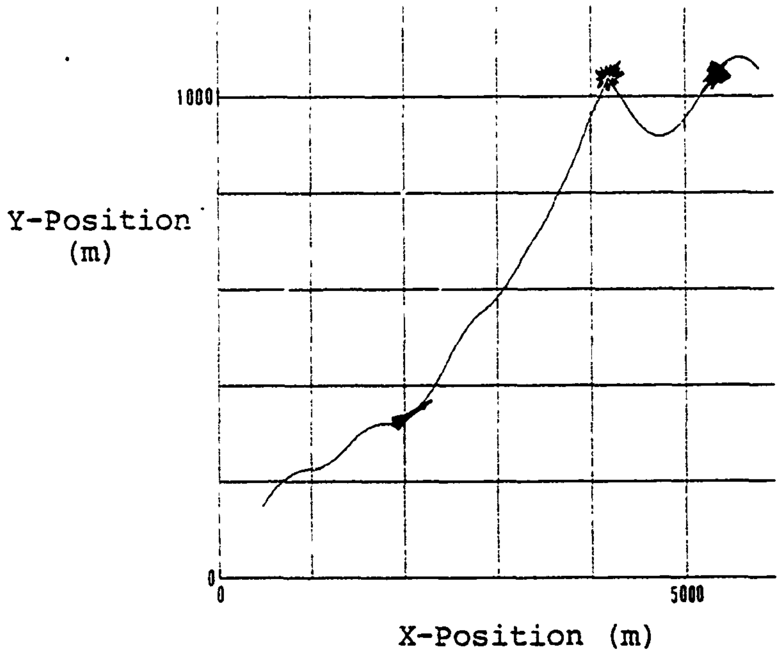


Figure VIII-6 'Consolidated' Simulation Missile & Target Flight Paths When Intermittently Jammed

When no jamming was encountered, the flight profile became more exaggerated because the target was maneuvering, thus causing the estimated impact point to move at an even higher rate. Recall that, in a non-jamming environment, the missile will "fly" for the impact point vice the target. Figure VIII-5 shows this profile. Finally, with intermittent jamming, the missile's flight profile, Figure VIII-6, became more erratic but did not reach the more extreme deviations experienced in the unjammed case. Therefore, it appears that the best method would be to use the "basic" command to line-of-sight guidance technique against a maneuvering target regardless of the jamming condition in effect. Also, as can be observed in Figure VIII-6, the flight profile during intermittent jamming, the amount of erratic variation in the missile's movement decreased considerably as it closed the target. This occurred because the difference between the synthetic line of sight and the line of sight becomes smaller with time until they are equal in value at the time of impact. Consequently, the "lead-angle" approach should be considered for utilization against a non-maneuvering target when adequate range information is available, i.e., when denial jamming is not encountered.

In order to reduce the amount of movement in the missile's thrust-vector controllers and, thus, to ensure that they do not "beat themselves to death" mechanically while trying to reduce a trivial cross-range error, a reduced amount of deflection was applied when the cross-range error was relatively small. This, then, was a trade-off of allowing some increase in cross-range error for improved thrust-vector control mechanical reliability. Figure VIII-7 shows how the cross-range error was reduced without using a smaller amount of thrust-vector control deflection. Keep in mind that the control surfaces in the missile were shifting rapidly from a total positive to a total negative direction. In Figure VIII-8, a cross-range error of four meters was accepted as a reasonable maximum miss distance and the amount of control was decreased when the error was within that value. This assured improved mechanical reliability of the system. As can be seen in Figure VIII-8,

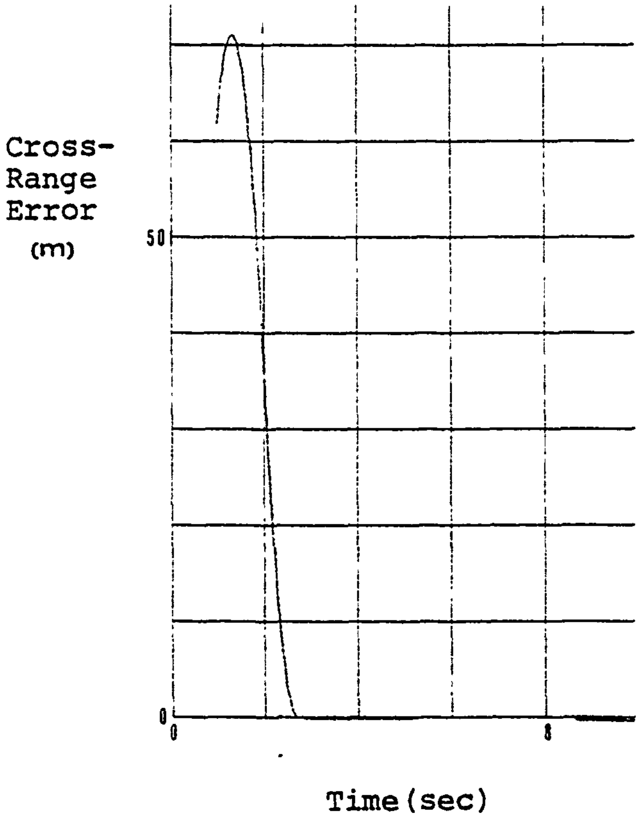


Figure VIII-7

'Consolidated' Simulation Missile Cross-Range Error With Full Missile Control Surface Deflections

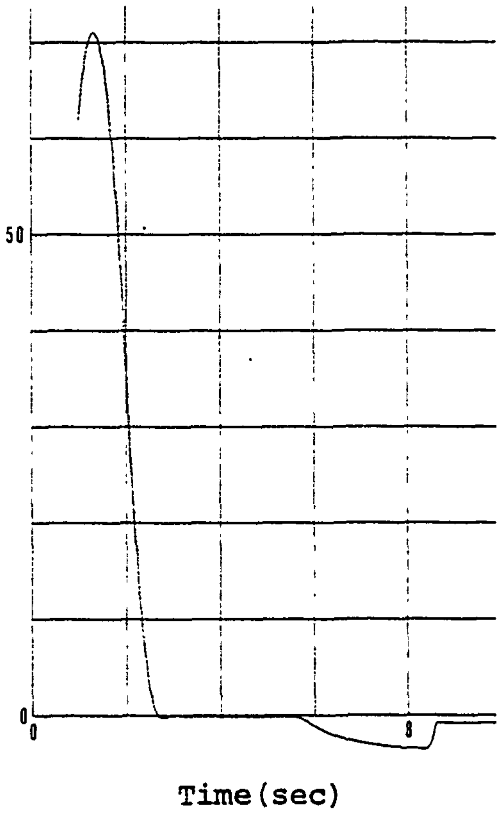


Figure VIII-8

'Consolidated' Simulation Missile Cross-Range Error With Reductions in Missile Control Surface Deflections

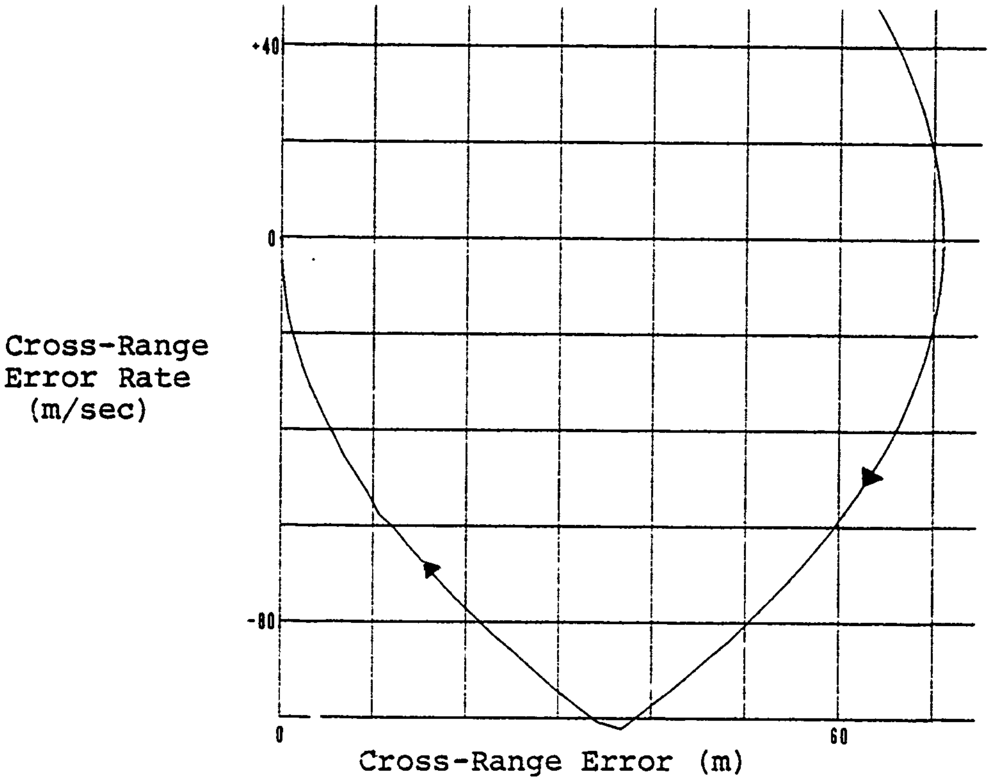


Figure VIII-9 'Consolidated' Simulation Phase Plane With Full Missile Control Surface Deflection

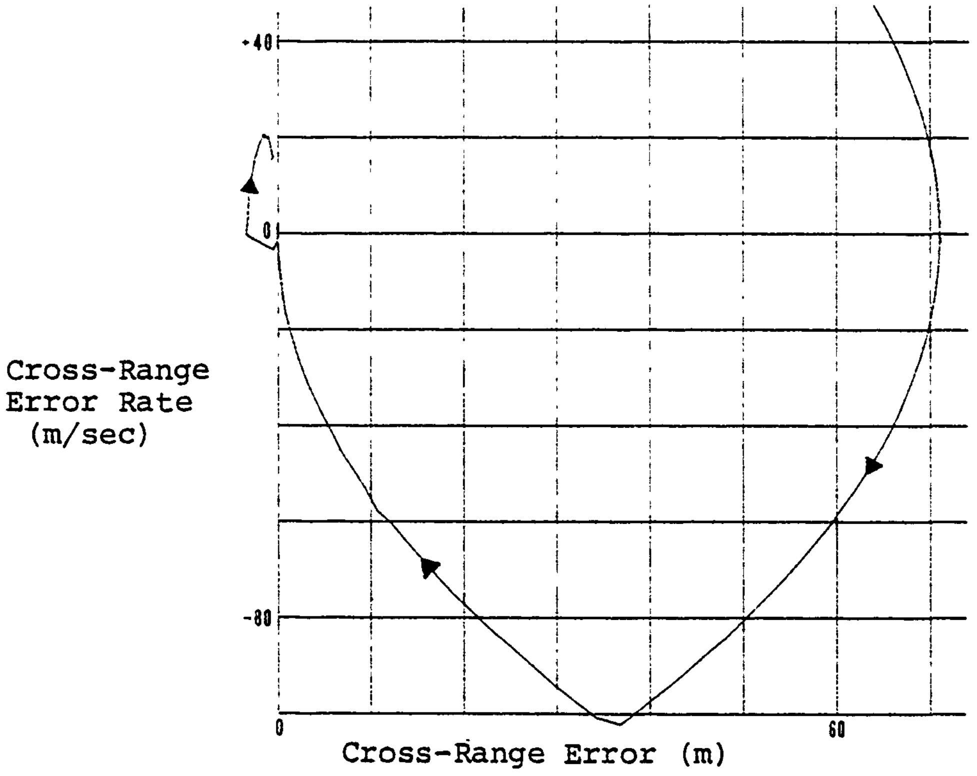


Figure VIII-10 'Consolidated' Simulation Phase Plane With Reductions in Missile Control Surface Deflections at the time of impact the cross-range error was in the process of being reduced from a value of about four meters. The phase planes, Figures VIII-9 and VIII-10, also, show the effects of this trade-off. Figure VIII-9 was the result obtained when using no reduction in the amount of thrust-vector control; while Figure VIII-10 shows the effect of allowing a maximum four-meter error with a reduced amount of lateral thrust.

# IX. CONCLUSIONS

These simulations support the premise that consideration should be given to utilizing a "lead-angle" variant in conjunction with the "basic" command to line-of-sight guidance technique. This would increase the missile system's effective engagement envelope against a non-maneuvering target that employs low-duty-cycle denial jamming. If the target utilizes higher duty cycle jamming or begins to make evasive maneuvers, then, the system can revert, when necessary, to the "basic" guidance mode even after the missile has been launched.

This new approach appears to be generically applicable to missiles of the command to line-of-sight guidance class. If a range-tracking capability already is contained within the system, it should be possible to implement this improved guidance technique through modifications to the existing guidance computer.

# APPENDIX A

## DESCRIPTION AND USE OF SUBROUTINE INTEG2

INTEG2 provides a simple means for a Runga-Kutta solution of ordinary differential equations. Input and output routines (including graphical point plots on the versatec plotter) are built in.

The subroutine has been compiled and added to the FORTRAN library, SYS1.MPSLIB. The program is flexible and self contained. It will seldom be desirable to build the subroutine into a more complex program.

It is assumed that the equations to be solved have been reduced to 'N' simultaneous first order differential equations of the form:

```
DX(I)/DT = F(X(1),X(2),...,X(N),T),  I = 1,2,...,N
```

where the function 'F' may be nonlinear. The user must translate these 'N' equations into FORTRAN as in the following second-order example:

```
XDOT(1)  = X(2)
XDOT(2)  = C(1)*X(1)-0.2*X(2)+C(2)*SIN(T*T)
```

C(1) and C(2) are constants which are entered on data cards.

To solve these equations the OS/360 deck must be of the following form:

```
//----STANDARD JOB CARD (GREEN)----
//  EXEC  FORTCLGV,REGION.GO=150K
//FORT.SYSIN  DD *
      DIMENSION X(30),XDOT(30),C(15)
```

```
  C(10) = 1.
1 CALL INTEG2(T,X,XDOT,C)
  . . .
  USERS FORTRAN EQUATIONS
  . . .
  GO TO 1
  END
  /*
  //GO.SYSIN  DD  *
  DATA CARDS AS DESCRIBED BELOW
  /* (ORANGE)
```

The number of XDOT equations is 'N', which must not exceed 30. These equations are supplied by the user, who defines each 'XDOT' in terms of the dependent variables X(1) through X(N) and the independent variable 'T'. In writing these equations the user may introduce at his convenience:

any unsubscripted variables;

the subscripted variables X(I), where the variable 'I' must be less than or equal to 30;

the constants C(I) to be entered as data, where the variable 'I' must be less than or equal to eight;

any normal FORTRAN technique or function; and

routines from any source library or user-supplied subroutines.

Note 1: The use of 'auxiliary' X(I), when the variable 'I' is greater than the number of equations to be integrated, does not alter the value of 'N', the order of the equations.

Note 2: Loops, either with or without a 'DO' statement, are best avoided.

Note 3: 'IF' statements, provided that they do not create a loop, can be used to transfer control within the user's equations. For example, the statement, IF(T.GT.10.)C(3)=0., would cause C(3) to take the value zero for all 'T' greater than ten.

The constants C(1) through C(8) may be used as described in the above examples, and are read in from a data card. C(10) must never be used, except as indicated in the standard OS/360 deck above. C(11) and C(12) control the output. For example, if the statements, C(11) = 10.0 and C(12) = 2.0, are added to the users FORTRAN equations, every tenth integration step will be printed out, and every second step will be plotted. (If not set by the user, default values of 20 and five apply.) C(13) can be similarly used to modify the step size of the numerical integration which is more usually defined on a data card.

Only the independent variable 'T' and the variables X(1) through X(30) can be output. Therefore, to output a quantity which is not one of the 'N' dependent variables. The user must introduce an 'auxiliary' variable X(I), where the variable 'I' must be less than 31 but greater than 'N' - for example, by introducing the FORTRAN equation, X(27) = X(3)*X(3), the square of X(3) can be output by the program as X(27). Normally, a line of printout is generated after every 20 integration steps, and one graph point (per curve) after every five integration steps. These values can be modified as indicated in the previous section. A run is terminated if the run exceeds 450 lines or if more than 900 graph points are generated per curve.

The accuracy of the numerical integration is controlled by the choice of step size. Unfortunately there is no simple method for making this choice other than by trial and error. As a first guess, a step size of 1/1000 of the total range of interest of the independent variable 'T' is usually reasonable. If one or more of the variables change rapidly in part of the range of 'T', then a finer step size may be necessary for all or part of the run. The only way to be reasonably sure of the solution accuracy is to re-run the problem with a smaller step size and, then, to compare the results. A maximum of 4500 integration steps is permitted in any one run.

Data cards are assembled as follows:

First: The user's job identification label is punched in columns 1-42.

Second: The number of runs to be processed, which must be less than ten, is punched in column one. The run number together with the user's job identification label is placed on all output.

Third: The order of the differential equations, which must be less than 31, is punched with two digits in columns one and two - for example, '03'.

Fourth: Initial and final values of the independent variable (TI and TF) and the step size (DT) appear in the order TI - DT - TF and are punched, with decimal points, in columns 1-10, 11-20 and 21-30. Also, it is possible to process the integration in either two or three segments of differing step size. The data, in columns of ten as above, then takes the form TI - DT1 - TF1 - DT2 - TF or TI - DT1 - TF1 - DT2 - TF2 -DT3 - TF.

Fifth: The values of the constants C(1), C(2),..... C(8) are punched, with decimal points, in columns 1-10, 11-20,...,71-80. Ten blank columns may be used for a zero or unused constant. If no constants are used, this entire card will be blank --- but it must not be omitted.

Sixth: The initial values of X(1), X(2),...,X(N) are punched, with decimal points, in columns 1-10,11-20,... . Additional cards are required if 'N' is greater than eight. Ten blank columns may be used for a zero initial value.

Seventh: This card controls the choice of variables for printout. Each group of ten columns, 1-10,11-20,...up to 71-80, may be used to specify a column heading (eight characters) and a two-digit, right-justified subscript identifying the corresponding variable - for example: TIME 00 and SPEED 03 in columns 1-10 and 11-20 would cause the independent variable 'T', represented on this card by the subscript 00, to be printed out under the column heading 'TIME' and the variable X(3) to be printed out under the heading 'SPEED'. No more than eight variables can be output during one run. If no printout is desired, this card must be blank.

Eighth: This card controls the choice of variables for graph output. Up to four curves can be plotted, either all on separate graphs or all on a single graph. Each group of 20 columns 1-20, 21-40,...specifies a curve - for example: SPEED VS. TIME 0300 (in columns 1-20) would cause X(3) to be plotted vertically against the independent variable 'T' horizontally. (Again, the subscript 00 represents the independent variable 'T'.) The graph output would be labeled 'SPEED VS. TIME'. Note that the first 16 columns of each group are used for the label, the seventeenth and eighteenth for the 'Y' ordinates, and the last two for the 'X' ordinates.

If the curves are to be on separate graphs, each graph must have a label. If the curves are to be all on one graph, only the first label, columns 1-16, must be provided. The other labels must be blank. The entire card must be blank if no graphical output is required.

If several solutions are required of the same equations but with different constants or initial conditions, the number of runs, specified on the second data card, can be increased. For all runs after the first, only the fourth through the last data cards are to be supplied. Except for the data on the first three cards, and the equations themselves, no information is retained between runs.

Caution should be exercised by the user because too large an integration step size can result in unacceptable errors in the solution, and even in instability in the integration process.

This subroutine is maintained for use with the Naval Postgraduate School, Monterey, California, IBM 360 computer. It was programmed by J. R. Ward in October, 1963; was revised in November, 1971; and was redocumented in October, 1973.

## COMPUTER PROGRAMS

## 'BASIC' COMMAND TO LINE-OF-SIGHT SIMULATION

```
C***** INSERT STANDARD GREEN JOB CARD WITH TIME=3 AT THIS POSITION
//  EXEC  FORTCLGV
//FORT.SYSIN DD *
C
C          BASIC CLOS MISSILE GUIDANCE SIMULATION
C
C
C
C          A       INITIAL X POSITION OF TARGET (METERS)
C          B       INITIAL Y POSITION OF TARGET (METERS)
C          C(1)    INPUT VALUE ASSIGNED TO FLAG K.  0 PRODUCES FULL
C                      PROGRAM OUTPUT WHILE 1 'FREEZES' THE PROGRAM
C                      OPERATION AT THE CLOSEST POINT OF
C                      APPROACH, CPA.  (UNITLESS)
C          C(10)   CONSTANT FOR SUBROUTINE INTEG2 WHICH ALWAYS IS
C                      SET WITH VALUE 1.0 AS REQUIRED BY THE
C                      SUBROUTINE. (UNITLESS)
C          C(11)   DETERMINES HOW OFTEN INTEGRATION STEPS ARE
C                      PRINTED OUT (IF EQUAL 5 THEN EVERY FIFTH
C                      PRINTED OUT) DEFAULT 20  (UNITLESS)
C          C(12)   DETERMINES HOW OFTEN INTEGRATION STEPS ARE
C                      PLOTTED (IF EQUAL 2 THEN EVERY SECOND
C                      STEP IS PLOTTED) DEFAULT 5 (UNITLESS)
C          E       ANGULAR ERROR. THE ANGULAR DIFFERENCE BETWEEN
C                      THE COURSE THE MISSILE SHOULD 'FLY' AND
C                      THE ANGLE TO THE MISSILE. (RADIANS)
C          EDOT    ANGULAR ERROR RATE (RADIANS PER SECOND)
C          EX      VARIABLE WHOSE SIGN(+-) DETERMINES MISSILE
C                      MANEUVER DIRECTION. THE ERROR FUNCTION (METERS)
C          EX1     GEOMETRIC DETERMINATION OF PERPENDICULAR DISTANCE
C                      OF MISSILE FROM THE LINE CF SIGHT, LOS. (METERS)
C          EX2     TIME RATE OF CHANGE OF EX1 (METERS PER SECOND)
C          G       MAGNITUDE OF THE MISSILE CROSS-RANGE
C                      ACCELERATION (METERS PER SECOND SQUARED)
C          K       FLAG USED TO INDICATE ATTAINMENT OF CPA AND TO
C                      'FREEZE' PROGRAM OPERATION (UNITLESS)
C          PI      THE VALUE OF PI. (UNITLESS)
C          RM      ESTIMATED RANGE TO MISSILE (METERS)
C          SIGM    ANGLE OF MISSILE FROM THE REFERENCE PLANE (RADIANS)
C          SIGT    ANGLE OF TARGET  LOS FROM REFERENCE PLANE (RADIANS)
```

```
C     T       TIME (SECONDS)
C     TT      PREVIOUS ITERATIVE VALUE OF TIME (SECONDS)
C     U       MISSILE CROSS-RANGE ACCELERATION USED IN STATE
C                  VARIABLE ANALYSIS, SVA, INCLUDING DIRECTION
C                  AS DETERMINED BY SIGN OF EX (METERS PER
C                  SECOND SQUARED)
C     VM      MISSILE VELOCITY (METERS PER SECOND)
C     VTX     TARGET VELOCITY IN X DIRECTION (METERS PER SECOND)
C     VTY     TARGET VELOCITY IN Y DIRECTION (METERS PER SECOND)
C     X(1)    DETERMINATION OF PERPENDICULAR DISTANCE OF
C                  MISSILE FROM  LOS, CROSS-RANGE ERROR (METERS)
C     X(2)    TIME RATE OF CHANGE OF X(1), CROSS-RANGE
C                  ERROR RATE (METERS PER SECOND)
C     X(7)    MISSILE Y POSITION (METERS)
C     X(8)    MISSILE X POSITION (METERS)
C     X(9)    ANGLE OF MISSILE FRCM THE REFERENCE PLANE (DEGREES)
C     X(10)   OUTPUT VARIABLE FOR U (METERS PER SECCND SQUARED)
C     X(11)   ANGLE OF TARGET LCS FROM REFERENCE PLANE (DEGREES)
C     X(12)   SEPARATION OF MISSILE FROM TARGET (METERS)
C     X(13)   OUTPUT VARIABLE FOR E     (DEGREES)
C     X(14)   OUTPUT VARIABLE FOR EDOT (DEGREES PER SECOND)
C     X(16)   OUTPUT VARIABLE FOR EX (METERS)
C     X(27)   TARGET Y POSITION (METERS)
C     X(28)   TARGET X POSITION (METERS)
C     XDOT(1) SVA EQUATION
C     XDOT(2) SVA EQUATION
C     Z       STORAGE FOR PREVICUS VALUE OF X(12). THE
C                  INITIAL VALUE IS LARGE TO ENSURE PROGRAM
C                  IS NOT PREMATURELY 'FROZEN'. (METERS)
C
C           * * * * * * * * * * * * * * * * * * * * * * * * *
C           * INITIALIZATION OF CONDITIONS PRIOR TO FIRST RUN *
C           * NOTE, IF ALTERED DURING A RUN, THESE CONDITIONS *
C           * ARE NOT RESET FOR SUBSEQUENT RUNS.              *
C           * * * * * * * * * * * * * * * * * * * * * * * * *
C
      DIMENSION X(30),XDOT(30),C(15)
      C(10)=1.
      C(11)=5.
      C(12)=2.
      PI=3.14159265
      VM=500.
      A=5000.
      B=0.
      SIGM=ATAN(B/A)
      VTX=(-177.)
      VTY=177.
      Z=100000.
      TT=0.
C
C              * * * * * * * * * * * * * * * * * * * * * * * * *
C              * SUBROUTINE FOR RUNGA-KUTTA SOLUTION OF ORDINARY *
C              * DIFFERENTIAL EQUATICNS.  NOTE, NEW VALUES FOR   *
C              * RUN TIMES, TIME INCREMENTS, C(N) CONSTANTS, AND *
C              * X(N) INITIAL CONDITIONS ARE READ.               *
C              * * * * * * * * * * * * * * * * * * * * * * * * *
C
    1 CALL INTEG2(T,X,XDOT,C)
C
C              * * * * * * * * * * * * * * * * * * * * * * * * *
C              * VALUE OF C(1) CHANGEC TO AN INTEGER STORED AS K *
C              * * * * * * * * * * * * * * * * * * * * * * * * *
C
      K=C(1)
C
C              * * * * * * * * * * * * * * * * * * * * * * * * *
C              * IF THE INTEGER VALUE OF C(1) IS 2, THE PROGRAM  *
C              * FREEZES TARGET AND MISSILE POSITIONS BY GCING   *
C              * INTO AN INFINITE LCCP UNTIL THE INTEGRATION     *
C              * REACHES FINAL TIME.                             *
C              * * * * * * * * * * * * * * * * * * * * * * * * *
C
      IF(K.EQ.2)GO TO 1
C
C              * * * * * * * * * * * * * * * * * * * * * * * * *
C              * TARGET X POSITION DETERMINED.                   *
C              * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(28)=A+VTX*T
C
C              * * * * * * * * * * * * * * * * * * * * * * * * *
C              * TARGET Y POSITION DETERMINED.                   *
C              * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(27)=B+VTY*T
C
C              * * * * * * * * * * * * * * * * * * * * * * * * *
C              * TARGET ANGLE FROM REFERENCE CALCULATED.         *
C              * * * * * * * * * * * * * * * * * * * * * * * * *
C
      SIGT=ATAN(X(27)/X(28))
C
C              * * * * * * * * * * * * * * * * * * * * * * * * *
C              * ESTIMATED RANGE FROM TRACKER TO MISSILE         *
C              * CALCULATED FROM TIME OF FLIGHT                  *
C              * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
C
      RM=VM*T
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * ANGULAR ERROR AND ANGULAR ERROR RATE DETERMINED *
C                  * NOTE, EDOT IS THE DERIVATIVE CF E.              *
C                  * * * * * * * * * * * * * * * * * * * * * * * * *
C
      E=SIGM-SIGT
      ECOT=(+1./SQRT(1.-X(1)*X(1)/(RM*RM)))*(X(2)/RM-X(1)/(RM*T))
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * GEOMETRIC DETERMINATION OF THE PERPENDICULAR    *
C                  * DISTANCE OF THE MISSILE FROM THE LINE OF SIGHT. *
C                  * CALCULATION OF THE RATE OF CHANGE IN THAT       *
C                  * DISTANCE.                                       *
C                  * * * * * * * * * * * * * * * * * * * * * * * * *
C
      EX1=RM*SIN(E)
      EX2=RM*EDOT*COS(E)+SIN(E)*VM
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * REDUCTION OF THE AMOUNT OF THRLST-VECTOR        *
C                  * CONTROL UTILIZED WHEN PHASE PLANE ERROR IS      *
C                  * CLOSE TO ZERO.                                  *
C                  * * * * * * * * * * * * * * * * * * * * * * * * *
C
      G=60.
      IF((ABS(EX1)+ABS(EX2)).LE.1.0)G=G/6.
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * PARABOLIC SWITCHING BOUNDARY ESTABLISHED WHICH  *
C                  * ZEROS ERROR IN MINIMUM TIME.                    *
C                  * * * * * * * * * * * * * * * * * * * * * * * * *
C
      EX=EX1+(EX2*ABS(EX2)/(2.*G))
      U=(-G)*SIGN(1.,EX)
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * STATE VARIABLE EQLATIONS                        *
C                  * * * * * * * * * * * * * * * * * * * * * * * * *
C
      XDOT(1)=X(2)
      XCOT(2)=U
C
```

```
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * CALCULATION OF REVISED ANGLE OF MISSILE FROM     *
C                  * REFERENCE                                        *
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      SIGM=SIGT+ARSIN(X(1)/RM)
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * MISSILE Y POSITION DETERMINED                    *
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(7)=RM*SIN(SIGM)
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * MISSILE X POSITION DETERMINED                    *
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(8)=RM*COS(SIGM)
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * VARIABLES PREPARED FOR OUTPUT                    *
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(9)=SIGM*180./PI
      X(10)=U
      X(11)=SIGT*180./PI
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * MISSILE-TO-TARGET RANGE CALCULATED               *
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(12)=SQRT((X(27)-X(7))*(X(27)-X(7))+(X(28)-X(8))*(X(28)-X(8)))
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * IF INTEGER VALUE FOR C(1) IS ZERO, DO NOT        *
C                  * FREEZE PROGRAM AT CLCSEST POINT OF APPROACH,     *
C                  * OTHERWISE, FREEZE PROGRAM, AFTER 5 SECONDS OF    *
C                  * FLIGHT, WHEN BOTH TIME AND MISSILE-TO-TARGET     *
C                  * RANGE ARE GREATER THAN THE PREVIOUS ITERATION    *
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      IF(K.EQ.0)GO TO 2
      IF(X(12).GT.Z.AND.T.GT.TT.AND.T.GT.5.0)C(1)=2.
C
```

```
C                  * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * STORE CURRENT VALUES OF TIME AND MISSILE-TO-    *
C                  * TARGET RANGE FOR CCMFARISON DURING THE NEXT     *
C                  * ITERATION.                                      *
C                  * * * * * * * * * * * * * * * * * * * * * * * * *
C
      Z=X(12)
      TT=T
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * VARIABLES PREPARED FOR OUTPUT                   *
C                  * * * * * * * * * * * * * * * * * * * * * * * * *
C
    2 X(13)=E*180./PI
      X(14)=EDOT*180./PI
      X(16)=EX
      X(20)=EX1
      X(21)=EX2
      GC T3 1
      END
C***** INSERT A CARD WITH /* IN COLUMNS ONE AND TWO AT THIS POSITION
//GO.FT06F001 DD SPACE=(CYL,(6,1))
//GO.SYSIN DD *
 BASIC SIMULATION
3
 2
1.005      .005       10.

10.        50.
TIME      00CRE     01CRERATE 02M-T SEP 12MX POSIT08TX POSIT28MY POSIT07TY POSIT27
POSITION VS TIME0800                0700                2800                270C
1.005      .005       10.

10.        50.
TIME      00SIGM    09SIGT    11EDOT    14EX1     20EX2     21EX      16U       10
 CRE VS TIME    0100 CRE RATE VS CRE0201
1.005      .005       10.
1.
10.        50.

RELATIVE POSITS 0708                2728
C***** INSERT /* CRANGE CARD AT THIS POSITION IN THE PROGRAM
```

## 'LEAD-ANGLE' COMMAND TO LINE-OF-SIGHT ULATION

```
C***** INSERT STANDARD GREEN JOB CARD WITH TIME=3 AT THIS POSITION
//   EXEC  FORTCLGV
//FORT.SYSIN DD *
C
C            LEAD-ANGLE CLOS MISSILE GUIDANCE SIMULATION
C
C            A        INITIAL X POSITION OF TARGET (METERS)
C            B        INITIAL Y POSITION OF TARGET (METERS)
C            C(1)     INPUT VALUE ASSIGNEC TO FLAG K.  O PRODUCES FULL
C                         PROGRAM OUTPUT WHILE 1 'FREEZES' THE PROGRAM
C                         OPERATION AT THE CLOSEST POINT OF
C                         APPROACH, CPA. (UNITLESS)
C            C(2)     INPUT VALUE ASSIGNEC TO FLAG L.  O, WHICH IS
C                         ALWAYS READ AS INPUT, INDICATES THAT THE
C                         MISSILE HAS NOT REACHED CPA TO THE TARGET.
C                         THE VALUE IS CHANGED TO 1 WHEN THE MISSILE
C                         REACHES CPA. (UNITLESS)
C            C(10)    CONSTANT FOR SUBROUTINE INTEG2 WHICH ALWAYS IS
C                         SET WITH VALUE 1.0 AS REQUIRED BY THE
C                         SUBROUTINE. (UNITLESS)
C            C(11)    DETERMINES HOW OFTEN INTEGRATION STEPS ARE
C                         PRINTED OUT (IF EQUAL 5 THEN EVERY FIFTH
C                         PRINTED OUT) DEFAULT 20   (UNITLESS)
C            C(12)    DETERMINES HOW OFTEN INTEGRATION STEPS ARE
C                         PLOTTED (IF EQUAL 2 THEN EVERY SECOND
C                         STEP IS PLOTTED) DEFAULT 5 (UNITLESS)
C            E        ANGULAR ERROR. THE ANGULAR DIFFERENCE BETWEEN
C                         THE COURSE THE MISSILE SHOULD 'FLY' AND
C                         THE ANGLE TO THE MISSILE. (RADIANS)
C            EDOT     ANGULAR ERROR RATE (RADIANS PER SECOND)
C            EX       VARIABLE WHOSE SIGN(+-) DETERMINES MISSILE
C                         MANEUVER DIRECTION. THE ERROR FUNCTION (METERS)
C            EX1      GEOMETRIC DETERMINATION OF PERPENDICULAR DISTANCE
C                         OF MISSILE FROM THE LINE OF SIGHT, LOS. (METERS)
C            EX2      TIME RATE OF CHANGE OF EX1 (METERS PER SECOND)
C            G        MAGNITUDE OF THE MISSILE CROSS-RANGE
C                         ACCELERATION (METERS PER SECOND SQUARED)
C            K        FLAG USED TO INDICATE ATTAINMENT OF CPA AND TO
C                         'FREEZE' PROGRAM OPERATION (UNITLESS)
C            L        FLAG USED TO INDICATE ATTAINMENT OF CPA ANC TO
C                         SHIFT THE SIGN OF TOFR. (UNITLESS)
C            PI       THE VALUE OF PI. (UNITLESS)
C            RM       RANGE TO MISSILE (METERS)
C            SIGM     ANGLE OF MISSILE FRCM THE REFERENCE PLANE (RADIANS)
```

```
C                SIGT    ANGLE OF TARGET LOS FROM REFERENCE PLANE (RADIANS)
C                SLOS    ANGLE OF SYNTHETIC LOS FROM REFERENCE (RADIANS)
C                T       TIME (SECONDS)
C                TOFR    TIME OF FLIGHT REMAINING UNTIL IMPACT (SECONDS)
C                TT      PREVIOUS ITERATIVE VALUE OF TIME (SECONDS)
C                U       MISSILE CROSS-RANGE ACCELERATION USED IN STATE
C                             VARIABLE ANALYSIS, SVA, INCLUDING DIRECTION
C                             AS DETERMINED BY SIGN OF EX (METERS PER
C                             SECOND SQUARED)
C                VC      MISSILE-TARGET CLOSING VELOCITY (METERS PER SECOND)
C                VM      MISSILE VELOCITY (METERS PER SECOND)
C                VMX     MISSILE VELOCITY IN X DIRECTION (METERS PER SECOND)
C                VMY     MISSILE VELOCITY IN Y DIRECTION (METERS PER SECOND)
C                VTX     TARGET VELOCITY IN X DIRECTION (METERS PER SECOND)
C                VTY     TARGET VELOCITY IN Y DIRECTION (METERS PER SECOND)
C                X(1)    DETERMINATION OF PERPENDICULAR DISTANCE OF
C                             MISSILE FROM LOS, CROSS-RANGE ERROR (METERS)
C                X(2)    TIME RATE OF CHANGE OF X(1), CROSS-RANGE
C                             ERROR RATE (METERS PER SECOND)
C                X(7)    MISSILE Y POSITION (METERS)
C                X(8)    MISSILE X POSITION (METERS)
C                X(9)    ANGLE OF MISSILE FROM THE REFERENCE PLANE (DEGREES)
C                X(10)   OUTPUT VARIABLE FOR U (METERS PER SECOND SQUARED)
C                X(11)   ANGLE OF TARGET LOS FROM REFERENCE PLANE (DEGREES)
C                X(12)   SEPARATION OF MISSILE FROM TARGET (METERS)
C                X(13)   OUTPUT VARIABLE FOR E     (DEGREES)
C                X(14)   OUTPUT VARIABLE FOR EDOT (DEGREES PER SECOND)
C                X(15)   OUTPUT VARIABLE FOR TOFR (SECONDS)
C                X(16)   OUTPUT VARIABLE FOR EX
C                X(17)   IMPACT POINT'S Y POSITION (METERS)
C                X(18)   IMPACT POINT'S X POSITION (METERS)
C                X(19)   OUTPUT VARIABLE FOR SLOS (DEGREES)
C                X(20)   OUTPUT VARIABLE FOR EX1 (METERS)
C                X(21)   OUTPUT VARIABLE FOR EX2 (METERS PER SECOND)
C                X(27)   TARGET Y POSITION (METERS)
C                X(28)   TARGET X POSITION (METERS)
C                XDOT(1) SVA EQUATION
C                XDOT(2) SVA EQUATION
C                W       STORAGE FOR PREVIOUS VALUE OF X(1)   (METERS)
C                WW      STORAGE FOR PREVIOUS VALUE OF X(2) (METERS PER SECOND)
C                Z       STORAGE FOR PREVIOUS VALUE OF X(12). THE
C                             INITIAL VALUE IS LARGE TO ENSURE PROGRAM
C                             IS NOT PREMATURELY 'FROZEN'. (METERS)
C
```

```
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * INITIALIZATION OF CONDITIONS PRIOR TO FIRST RUN *
C                    * NOTE, IF ALTERED DURING A RUN, THESE CONDITIONS *
C                    * ARE NOT RESET FOR SUBSEQUENT RUNS.              *
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      DIMENSION X(30),XDOT(30),C(15)
      C(10)=1.
      C(11)=5.
      C(12)=2.
      PI=3.14159265
      VM=500.
      A=5000.
      B=0.
      SIGM=ATAN(B/A)
      VTX=(-177.)
      VTY=177.
      Z=100000.
      TT=0.
      TCFR=0.
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * SUBROUTINE FOR RUNGA-KUTTA SOLUTION OF ORDINARY *
C                    * DIFFERENTIAL EQUATICNS.  NOTE, NEW VALUES FOR   *
C                    * RUN TIMES, TIME INCREMENTS, C(N) CONSTANTS, AND *
C                    * X(N) INITIAL CONDITIONS ARE READ.               *
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C
    1 CALL  INTEG2(T,X,XDOT,C)
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * VALUE OF C(1) CHANGED TO AN INTEGER STORED AS K *
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      K=C(1)
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * IF CPA ATTAINED, FREEZE VALUES OF CROSS-        *
C                    * RANGE ERROR AND CRCSS-RANGE ERROR RATE          *
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      IF(K.EQ.2)X(1)=W
      IF(K.EQ.2)X(2)=WW
C
```

```
C
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C               * IF THE INTEGER VALUE OF C(1) IS 2, THE PROGRAM   *
C               * FREEZES TARGET AND MISSILE POSITIONS BY GOING    *
C               * INTO AN INFINITE LOOP UNTIL THE INTEGRATION      *
C               * REACHES FINAL TIME.                              *
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      IF(K.EQ.2)GO TO 1
C
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C               * TARGET X POSITION DETERMINED.                    *
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(28)=A+VTX*T
C
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C               * TARGET Y POSITION DETERMINED.                    *
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(27)=B+VTY*T
C
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C               * TARGET ANGLE FROM REFERENCE CALCULATED.          *
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      SIGT=ATAN(X(27)/X(28))
C
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C               * X POSITION OF IMPACT DETERMINED.                 *
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(18)=X(28)+VTX*TCFR
C
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C               * Y POSITION OF IMPACT DETERMINED.                 *
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(17)=X(27)+VTY*TCFR
C
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C               * SYNTHETIC LOS FROM REFERENCE CALCULATED          *
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      SLOS=ATAN(X(17)/X(18))
C
```

```
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C                   * ESTIMATED RANGE FRCM TRACKER TO MISSILE           *
C                   * CALCULATED FROM TIME OF FLIGHT                    *
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      RM=VM*T
C
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C                   * ANGULAR ERROR AND ANGULAR ERRCR RATE DETERMINED *
C                   * NOTE, EDOT IS THE DERIVATIVE OF E.              *
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      E=SIGM-SLOS
      EDOT=(+1./SCRT(1.-X(1)*X(1)/(RM*RM)))*(X(2)/RM-X(1)/(RM*T))
C
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C                   * GEOMETRIC DETERMINATION OF THE PERPENDICULAR    *
C                   * DISTANCE OF THE MISSILE FROM THE LINE CF SIGHT. *
C                   * CALCULATION OF THE RATE OF CHANGE IN THAT       *
C                   * DISTANCE.                                       *
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      EX1=RM*SIN(E)
      EX2=RM*EDOT*COS(E)+SIN(E)*VM
C
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C                   * REDUCTION OF THE AMOUNT OF THRUST-VECTOR        *
C                   * CONTROL UTILIZED WhEN PHASE PLANE ERROR IS      *
C                   * CLOSE TO ZERO.                                  *
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      G=60.
      IF((ABS(EX1)+ABS(EX2)).LE.1.0)G=G/6.
C
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C                   * PARABOLIC SWITCHING BOUNDARY ESTABLISHED WHICH  *
C                   * ZEROS ERROR IN MINIMUM TIME.                    *
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      EX=EX1+(EX2*ABS(EX2)/(2.*G))
      U=(-G)*SIGN(1.,EX)
C
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C                   * STATE VARIABLE EQUATIONS                        *
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      XDOT(1)=X(2)
      XDOT(2)=U
```

```
C
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C               * CALCULATION OF REVISED ANGLE OF MISSILE FROM      *
C               * REFERENCE                                         *
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      SIGM=SLOS+ARSIN(X(1)/RM)
C
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C               * MISSILE Y POSITION DETERMINED                     *
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(7)=RM*SIN(SIGM)
C
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C               * MISSILE X POSITION DETERMINED                     *
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(8)=RM*COS(SIGM)
C
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C               * VARIABLES PREPARED FOR OUTPUT                     *
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(9)=SIGM*180./PI
      X(10)=U
      X(11)=SIGT*180./PI
C
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C               * MISSILE-TO-TARGET RANGE CALCULATED                *
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(12)=SQRT((X(27)-X(7))*(X(27)-X(7))+(X(28)-X(8))*(X(28)-X(8)))
C
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C               * MISSILE VELOCITY IN X DIRECTICN DETERMINED        *
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      VMX=VM*COS(SIGM)
C
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C               * MISSILE VELOCITY IN Y DIRECTICN DETERMINED        *
C               * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      VMY=VM*SIN(SIGM)
C
```

```
C                    * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * MISSILE-TO-TARGET CLCSING VELOCITY CALCULATED   *
C                    * * * * * * * * * * * * * * * * * * * * * * * * *
C
      VC=SQRT((VTY-VMY)*(VTY-VMY)+(VTX-VMX)*(VTX-VMX))
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * TIME OF FLIGHT REMAINING UNTIL IMPACT           *
C                    * COMPUTED.                                       *
C                    * * * * * * * * * * * * * * * * * * * * * * * * *
C
      TOFR=X(12)/VC
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * FREEZE PROGRAM, AFTER 5 SECONDS OF FLIGHT,      *
C                    * WHEN BOTH TIME AND MISSILE-TO-TARGET RANGE ARE  *
C                    * GREATER THAN THE PREVIOUS ITERATION.            *
C                    * MISSILE HAS REACHED CPA.                        *
C                    * * * * * * * * * * * * * * * * * * * * * * * * *
C
      IF(X(12).GT.Z.AND.T.GT.TT.AND.T.GT.5.0)C(1)=2.
      K=C(1)
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * IF CPA ATTAINED, TIME OF FLIGHT REMAINING       *
C                    * UNTIL IMPACT IS ZERO.                           *
C                    * * * * * * * * * * * * * * * * * * * * * * * * *
C
      IF(K.EQ.2)TOFR=0.
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * STORE CURRENT VALUES OF TIME, CRE, CRE RATE,    *
C                    * AND MISSILE-TO-TARGET RANGE FOR USE DURING THE  *
C                    * NEXT ITERATION.                                 *
C                    * * * * * * * * * * * * * * * * * * * * * * * * *
C
    2 Z=X(12)
      TT=T
      W=X(1)
      WW=X(2)
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * VARIABLES PREPARED FOR OUTPUT                   *
C                    * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(13)=E*180./PI
      X(14)=EDOT*180./PI
      X(15)=TOFR
      X(16)=EX
      X(19)=SLOS*180./PI
      X(20)=EX1
      X(21)=EX2
      GO TO 1
      END
C***** INSERT A CARD WITH /* IN COLUMNS ONE AND TWO AT THIS POSITION
//GO.FT06F001 DD SPACE=(CYL,(6,1))
//GO.SYSIN DD *
 LEAD-ANGLE SIMULATION
3
 2
1.005      .005      10.

10.        50.
TIME    00CRE     01CRERATE 02M-T SEP 12MX POSIT08TX POSIT28MY PCSIT07TY POSIT27
POSITION VS TIME0800                0700                2800                2700
1.005      .005      10.

10.        50.
TIME    00SIGM    09SIGT    11SLOS    19EX1     20EX2     21TOFR    15U       10
 CRE VS TIME    0100 CRE RATE VS CRE0201
1.005      .005      10.
1.
10.        50.

RELATIVE POSITS 0708                2728
C***** INSERT /* CRANGE CARD AT THIS POSITION IN THE PROGRAM
```

## 'CONSOLIDATED' COMMAND TO LINE-OF-SIGHT SIMULATION

```
C***** INSERT STANDARD GREEN JOB CARD WITH TIME=3 AT THIS POSITION
//   EXEC  FORTCLGV,REGION.PLOTGEN=250K
//FORT.SYSIN DD *
C
C             CONSOLIDATED CLOS MISSILE GUIDANCE SIMULATION
C
C
C
C             A       INITIAL X POSITION OF TARGET (METERS)
C             AC      MISSILE-TARGET CLOSING ACCELERATION MAGNITUDE (METERS
C                          PER SECOND SQUARED)
C             B       INITIAL Y POSITION OF TARGET (METERS)
C             C(1)    INPUT VALUE ASSIGNED TO FLAG K.  0 PRODUCES FULL
C                          PROGRAM OUTPUT WHILE 2 'FREEZES' THE PROGRAM
C                          OPERATION AT THE CLOSEST POINT OF
C                          APPROACH, CPA. (UNITLESS)
C             C(10)   CONSTANT FOR SUBROUTINE INTEG2 WHICH ALWAYS IS
C                          SET WITH VALUE 1.0 AS REQUIRED BY THE
C                          SUBROUTINE. (UNITLESS)
C             C(11)   DETERMINES HOW OFTEN INTEGRATION STEPS ARE
C                          PRINTED OUT (IF EQUAL 5 THEN EVERY FIFTH
C                          PRINTED OUT) DEFAULT 20  (UNITLESS)
C             C(12)   DETERMINES HOW OFTEN INTEGRATION STEPS ARE
C                          PLOTTED (IF EQUAL 2 THEN EVERY SECOND
C                          STEP IS PLOTTED) DEFAULT 5 (UNITLESS)
C             D       HALF A PERIOD OF TARGET'S SINUSOIDAL MOTION (SECONDS)
C             E       ANGULAR ERROR. THE ANGULAR DIFFERENCE BETWEEN
C                          THE COURSE THE MISSILE SHOULD 'FLY' AND
C                          THE ANGLE TO THE MISSILE. (RADIANS)
C             EDOT    ANGULAR ERROR RATE (RADIANS PER SECOND)
C             EX      VARIABLE WHOSE SIGN(+-) DETERMINES MISSILE
C                          MANEUVER DIRECTION. THE ERROR FUNCTION (METERS)
C             EX1     CROSS-RANGE ERROR (METERS)
C             EX2     CROSS-RANGE ERROR RATE (METERS PER SECOND)
C             G       MAGNITUDE OF THE MISSILE CROSS-RANGE
C                          ACCELERATION (METERS PER SECOND SQUARED)
C             I       COUNTER FOR 'DO' LOOPS (UNITLESS)
C             J       FLAG USED TO INDICATE WHETHER TRACKER IS JAMMED,
C                          INDICATED BY 1, OR NOT JAMMED, INDICATED BY 0.
C                          (UNITLESS)
C             K       FLAG USED TO INDICATE ATTAINMENT OF CPA AND TO
C                          'FREEZE' PROGRAM OPERATION (UNITLESS)
```

```
C        M         FLAG USED TO INDICATE WHETHER THE IMPACT
C                       POINT IS MOVING TOO FAST TO ATTACK,
C                       INDICATED BY 1, OR SUFFICIENTLY STABLE
C                       TO ATTACK, INDICATED BY 0. (UNITLESS)
C        P         STORAGE VARIABLE FOR ARITHMETIC EXPRESSION USED
C                       IN SUBSEQUENT CALCULATIONS IN THE PROGRAM, (UNITLESS)
C        PI        THE VALUE OF PI. (UNITLESS)
C        RM        ESTIMATED RANGE TO MISSILE (METERS)
C        SGMD      DERIVATIVE OF SIGM. RATE OF CHANGE IN
C                       THE MISSILE'S ANGULAR POSITION. (RADIANS PER
C                       SECOND)
C        SIGM      ANGLE OF MISSILE FROM THE REFERENCE PLANE (RADIANS)
C        SIGT      ANGLE OF TARGET LOS FROM REFERENCE PLANE (RADIANS)
C        SIGTD     TIME RATE OF CHANGE OF SIGT (RADIANS PER SECOND)
C        SLOS      ANGLE OF SYNTHETIC LOS FROM REFERENCE (RADIANS)
C        SLOSD     TIME RATE OF CHANGE OF SLOS (RADIANS PER SECOND)
C        T         TIME (SECONDS)
C        TJB       ARRAY USED TO STORE TIMES JAMMING BEGINS (SECONDS)
C        TJC       ARRAY USED TO STORE TIMES JAMMING CEASES (SECONDS)
C        TOFR      TIME OF FLIGHT REMAINING UNTIL IMPACT (SECONDS)
C        TOFRD     RATE OF CHANGE OF TIME OF FLIGHT REMAINING UNTIL
C                       IMPACT.      (UNITLESS)
C        TT        PREVIOUS ITERATIVE VALUE OF TIME (SECONDS)
C        U         MISSILE Y-DIRECTION ACCELERATION USED IN STATE
C                       VARIABLE ANALYSIS, SVA, INCLUDING DIRECTION
C                       AS DETERMINED BY SIGN OF EX (METERS PER
C                       SECOND SQUARED)
C        VC        MISSILE-TARGET CLOSING VELOCITY (METERS PER SECOND)
C        VI        MAGNITUDE OF THE IMPACT POINT'S VELOCITY.
C                       (METERS PER SECOND)
C        VIX       VELOCITY OF IMPACT POINT IN X DIRECTION
C                       (METERS PER SECOND)
C        VIY       VELOCITY OF IMPACT POINT IN Y DIRECTION
C                       (METERS PER SECOND)
C        VM        MISSILE VELOCITY (METERS PER SECOND)
C        VMX       MISSILE VELOCITY IN X DIRECTION (METERS PER SECOND)
C        VMXD      MISSILE X-DIRECTION ACCELERATION (METERS PER
C                       SECOND SQUARED)
C        VMY       MISSILE VELOCITY IN Y DIRECTION (METERS PER SECOND)
C        VMYD      MISSILE Y-DIRECTION ACCELERATION (METERS PER
C                       SECOND SQUARED)
C        VTX       TARGET VELOCITY IN X DIRECTION (METERS PER SECOND)
C        VTXD      TARGET X-DIRECTION ACCELERATION (METERS PER
C                       SECOND SQUARED)
C        VTXM      MAXIMUM TARGET VELOCITY IN THE X DIRECTION (METERS
C                       PER SECOND)
C        VTY       TARGET VELOCITY IN Y DIRECTION (METERS PER SECOND)
```

```
C          VTYD    TARGET Y-DIRECTION ACCELERATICN (METERS PER
C                        SECCND SQUARED)
C          VTYM    MAXIMUM TARGET VELOCITY IN THE Y DIRECTION (METERS
C                        PER SECOND)
C          X(1)    Y-DIRECTION DISTANCE OF MISSILE FROM REFERENCE,
C                        THE X-AXIS. (METERS)
C          X(2)    TIME RATE OF CHANGE OF X(1). (METERS PER SECOND)
C          X(3)    OUTPUT VARIABLE FOR VMX (METERS PER SECOND)
C          X(4)    OUTPUT VARIABLE FOR VMXD (METERS PER SECOND
C                        SQUARED)
C          X(5)    OUTPUT VARIABLE FCR VIX (METERS PER SECOND)
C          X(6)    OUTPUT VARIABLE FOR VIY (METERS PER SECOND)
C          X(7)    MISSILE Y POSITION (METERS)
C          X(8)    MISSILE X POSITION (METERS)
C          X(9)    ANGLE OF MISSILE FROM THE REFERENCE PLANE (DEGREES)
C          X(10)   OUTPUT VARIABLE FOR U (METERS PER SECOND SQUARED)
C          X(11)   ANGLE OF TARGET LOS FROM REFERENCE PLANE (DEGREES)
C          X(12)   SEPARATION OF MISSILE FROM TARGET (METERS)
C          X(13)   OUTPUT VARIABLE FOR E     (DEGREES)
C          X(14)   OUTPUT VARIABLE FOR EDOT (DEGREES PER SECOND)
C          X(15)   OUTPUT VARIABLE FOR TOFR (SECCNDS)
C          X(16)   OUTPUT VARIABLE FOR VI (METERS PER SECOND)
C          X(17)   IMPACT POINT'S Y POSITION (METERS)
C          X(18)   IMPACT POINT'S X POSITION (METERS)
C          X(19)   OUTPUT VARIABLE FOR SLOS (DEGREES)
C          X(20)   OUTPUT VARIABLE FOR EX1 (METERS)
C          X(21)   OUTPUT VARIABLE FOR EX2 (METERS PER SECOND)
C          X(22)   OUTPUT VARIABLE FOR J (UNITLESS)
C          X(23)   OUTPUT VARIABLE FCR SIGTD (DEGREES PER SECOND)
C          X(24)   OUTPUT VARIABLE FOR SLOSD (DEGREES PER SECOND)
C          X(25)   OUTPUT VARIABLE FOR M (UNITLESS)
C          X(26)   OUTPUT VARIABLE FOR AC (METERS PER SECCND SQUARED)
C          X(27)   TARGET Y POSITION (METERS)
C          X(28)   TARGET X POSITION (METERS)
C          X(29)   OUTPUT VARIABLE FOR TOFRD (UNITLESS)
C          XDOT(1) SVA EQUATION
C          XDOT(2) SVA EQUATION
C          XRATE   TIME RATE OF CHANGE CF X(12). THE DERIVATIVE OF
C                        X(12). (METERS PER SECOND)
C          X7      VARIABLE WHICH STCRES CURRENT VALUE OF X(27)
C                        AND ALLOWS X(27) TO BE INITIALIZED BETWEEN
C                        RUNS. (METERS)
C          X8      VARIABLE WHICH STORES CURRENT VALUE CF X(28)
C                        AND ALLOWS X(27) TO BE INITIALIZED BETWEEN
C                        RUNS. (METERS)
C          W       STORAGE FOR PREVIOUS VALUE OF X(1)    (METERS)
C          WW      STORAGE FOR PREVIOUS VALUE OF X(2) (METERS PFR SECOND)
C          Z       STORAGE FOR PREVICUS VALJE OF X(12)    (METERS)
```

```
C
C                 * * * * * * * * * * * * * * * * * * * * * * * * * *
C                 * INITIALIZATION OF CONDITIONS PRIOR TO FIRST RUN *
C                 * NOTE, IF ALTERED DURING A RUN, THESE CONDITIONS *
C                 * ARE NOT RESET FOR SUBSEQUENT RUNS.              *
C                 * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      DIMENSION X(30),XDOT(30),C(15),TJB(5),TJC(5)
 9500 FORMAT(2F10.7)
      C(10)=1.
      C(11)=5.
      C(12)=2.
      PI=3.14159265
      VM=500.
      SIGM=0.
      A=6000.
      B=1000.
      D=4.
      VTYM=64.
      VTXM=(-210.)
      TOFR=0.
      SLOSD=0.
      J=0
      M=0
      Z=0.
      TT=0.
      X7=0.
      X8=0.
      VMY=0.
      VMYD=0.
      VMX=0.
      VMXD=0.
      SGMD=0.
C
C                 * * * * * * * * * * * * * * * * * * * * * * * * * *
C                 * READ VALUES FOR JAMMING TIME INTERVALS          *
C                 * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      DO 10 I=1,5
      READ 9500,TJB(I),TJC(I)
   10 CONTINUE
C
C                 * * * * * * * * * * * * * * * * * * * * * * * * * *
C                 * SUBROUTINE FOR RUNGA-KUTTA SOLUTION OF ORDINARY *
C                 * DIFFERENTIAL EQUATIONS.  NOTE, NEW VALUES FOR   *
C                 * RUN TIMES, TIME INCREMENTS, C(N) CONSTANTS, AND *
C                 * X(N) INITIAL CONDITIONS ARE READ.               *
C                 * * * * * * * * * * * * * * * * * * * * * * * * * *
C
```

```
    1 CALL INTEG2(T,X,XDOT,C)
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * VALUE OF C(1) CHANGED TO AN INTEGER STORED AS K *
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      K=C(1)
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * IF CPA NOT ATTAINED, CONTINUE CALCULATICNS      *
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      IF(K.NE.2)GO TO 200
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * INITIALIZE CONDITIONS JUST PRICR TO NEXT RUN    *
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      IF(T.LT.9.9)GO TO 100
      J=0
      M=0
      SIGM=0.
      TOFR=0.
      SLCSD=0.
      X7=0.
      X8=0.
      VMY=0.
      VMYD=0.
      VMX=0.
      VMXD=0.
      SGMD=0.
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * HOLD VALUES OF X(1) AND X(2) CCNSTANT AFTER     *
C                    * PROGRAM FROZEN.                                 *
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C
  100 X(1)=W
      X(2)=WW
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * IF THE INTEGER VALUE OF C(1) IS 2, THE PROGRAM  *
C                    * FREEZES TARGET AND MISSILE POSITIONS BY GOING   *
C                    * INTO AN INFINITE LOOP UNTIL THE INTEGRATION     *
C                    * REACHES FINAL TIME.                             *
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      GO TO 1
```

```
C
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C                   * ESTIMATED RANGE FROM TRACKER TO MISSILE         *
C                   * CALCULATED FROM TIME OF FLIGHT                  *
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C
  20C RM=VM*T
C
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C                   * INTRODUCE ZERO INITIALIZATION BETWEEN RUNS FOR  *
C                   * VARIABLES X(7) AND X(8). AT OTHER TIMES THESE   *
C                   * FOLLOWING EQUATIONS HAVE NO EFFECT BECAUSE      *
C                   * THE VARIABLES WERE SET EQUAL CN THE PREVIOUS    *
C                   * ITERATION.                                      *
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(7)=X7
      X(8)=X8
C
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C                   * DETERMINE VELOCITIES AND ACCELERATIONS OF THE   *
C                   * TARGET                                          *
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      VTY=VTYM*COS(PI*T/D)
      VTYD=PI/D*VTYM*(-SIN(PI*T/D))
      VTX=VTXM
      VTXD=0.
C
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C                   * DETERMINE IF JAMMING OF THE TRACKING UNIT HAS   *
C                   * JUST BEGUN OR CEASED. SET J TO PROPER VALUE.    *
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      J=0
      DO 20 I=1,5
      IF(T.GE.TJB(I).AND.T.LT.TJC(I)) J=1
   20 CONTINUE
C
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C                   * TARGET X POSITION CETERMINED.                   *
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(28)=A+VTX*T
C
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
C                   * TARGET Y POSITION DETERMINED.                   *
C                   * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
C
      X(27)=B+VTYM*D/PI*SIN(PI*T/D)
C
C           * * * * * * * * * * * * * * * * * * * * * * * * * *
C           * TARGET ANGLE FROM REFERENCE AND TARGET ANGULAR   *
C           * RATE CALCULATED.                                 *
C           * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      SIGT=ATAN(X(27)/X(28))
      SIGTD=(1./(X(28)*X(28)+X(27)*X(27)))*(X(28)*VTY-X(27)*VTX)
C
C           * * * * * * * * * * * * * * * * * * * * * * * * * *
C           * X POSITION OF IMPACT REVISED.                    *
C           * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(18)=X(28)+VTX*TOFR
C
C           * * * * * * * * * * * * * * * * * * * * * * * * * *
C           * Y POSITION OF IMPACT REVISED.                    *
C           * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(17)=X(27)+VTY*TOFR
C
C           * * * * * * * * * * * * * * * * * * * * * * * * * *
C           * SYNTHETIC LOS FROM REFERENCE CALCULATED          *
C           * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      SLOS=ATAN(X(17)/X(18))
C
C           * * * * * * * * * * * * * * * * * * * * * * * * * *
C           * MISSILE-TO-TARGET RANGE CALCULATED               *
C           * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(12)=SQRT((X(27)-X(7))*(X(27)-X(7))+(X(28)-X(8))*(X(28)-X(8)))
C
C           * * * * * * * * * * * * * * * * * * * * * * * * * *
C           * MISSILE-TO-TARGET CLCSING VELOCITY CALCULATED    *
C           * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      VC=SQRT((VTY-VMY)*(VTY-VMY)+(VTX-VMX)*(VTX-VMX))
C
C           * * * * * * * * * * * * * * * * * * * * * * * * * *
C           * TIME OF FLIGHT REMAINING UNTIL IMPACT            *
C           * COMPUTED.                                        *
C           * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      TOFR=X(12)/VC
C
```

```
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * PREVENT POSSIBLE DIVISION BY ZERO.                *
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      IF(X(12).LT.0.000001)X(12)=.000001
      IF(VC.LT.0.000001)VC=.000001
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * DETERMINE VALUE OF F AND CALCULATE RATE OF       *
C                    * CHANGE OF X(12), XRATE. THIS IS USED LATER IN    *
C                    * DETERMINING TOFRD.                               *
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      P=SQRT(RM*RM-X(1)*X(1))
      XRATE=((X(27)-X(7))*(VTY-X(2))+(X(28)-X(8))*(VTX-RM*VM/P+X(1)*X(2)
     1/P))/X(12)
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * CALCULATE MISSILE-TARGET CLOSING ACCELERATICN    *
C                    * AND RATE OF CHANGE OF TOFR                       *
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      AC=((VTY-VMY)*(VTYD-VMYD)+(VTX-VMX)*(VTXD-VMXD))/VC
      TOFRD=XRATE/VC-X(12)*AC/(VC*VC)
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * DETERMINE RATE OF CHANGE IN THE IMPACT           *
C                    * POSITION AND SYNTHETIC LINE OF SIGHT.            *
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      VIX=VTX*(1.+TOFRD)+VTXD*TOFR
      VIY=VTY*(1.+TOFRD)+VTYD*TOFR
      VI=SQRT(VIX*VIX+VIY*VIY)
      SLOSD=(X(18)*VIY-X(17)*VIX)/(X(18)*X(18)+X(17)*X(17))
C
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C                    * DETERMINE IF TARGET MANEUVERS ARE CAUSING THE    *
C                    * PREDICTED IMPACT POINT TO MOVE AT TCC HIGH A     *
C                    * RATE FOR USE. IF SO, SET FLAG M TO 1 TO          *
C                    * INDICATE A HIGHLY MANEUVERING TARGET AND, THUS,  *
C                    * TO CAUSE THE MISSILE TO BE CONMANDED TC 'FLY'    *
C                    * DIRECTLY TOWARD THE TARGET VICE THE IMPACT       *
C                    * POINT.                                           *
C                    * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      M=0
      IF(VI.GT.(.8*VM))M=1
C
```

```
C
C                 * * * * * * * * * * * * * * * * * * * * * * * * *
C                 * CALCULATION OF ERROR AND ERROR RATE DEPENDING   *
C                 * UPON THE JAMMING CONDITION AND THE TARGET       *
C                 * MANEUVERING CONDITION. IF EITHER JAMMING OR     *
C                 * EXCESSIVE TARGET MANUEVERS ARE DETECTED THE     *
C                 * MISSILE SHOULD 'FLY' DIRECTLY TOWARC THE        *
C                 * TARGET. OTHERWISE, IT 'FLYS' DIRECTLY TOWARD    *
C                 * THE ESTIMATED IMPACT POSITION.                  *
C                 * * * * * * * * * * * * * * * * * * * * * * * * *
C
      IF(J.EQ.1.OR.M.EQ.1)GO TO 300
      E=SIGM-SLOS
      EDOT=SGMD-SLOSD
      GO TO 400
  3CC E=SIGM-SIGT
      EDOT=SGMD-SIGTD
C
C                 * * * * * * * * * * * * * * * * * * * * * * * * *
C                 * CALCULATION OF CROSS-RANGE ERROR AND ERROR RATE *
C                 * * * * * * * * * * * * * * * * * * * * * * * * *
C
  400 EX1=RM*SIN(E)
      EX2=RM*EDOT*COS(E)+SIN(E)*VM
C
C                 * * * * * * * * * * * * * * * * * * * * * * * * *
C                 * REDUCTION OF THE AMOUNT OF THRUST-VECTOR        *
C                 * CONTROL UTILIZED WHEN PHASE PLANE ERROR IS      *
C                 * CLOSE TO ZERO.                                  *
C                 * * * * * * * * * * * * * * * * * * * * * * * * *
C
      G=150.
      IF((ABS(EX1)+ABS(EX2)).LE.4.0)G=G/6.
C
C                 * * * * * * * * * * * * * * * * * * * * * * * * *
C                 * PARABOLIC SWITCHING BOUNDARY ESTABLISHED WHICH  *
C                 * ZEROS ERROR IN MINIMUM TIME. VALUE FCR THE      *
C                 * ERROR FUNCTION IS CALCULATED DEPENDING UPCN     *
C                 * WHETHER TARGET IS JAMMING, MANEUVERING, OR NOT. *
C                 * DETERMINE THE MISSILE'S Y-DIRECTION             *
C                 * ACCELERATION WHICH IS THE FORCING FUNCTION FOR  *
C                 * THE STATE VARIABLE ECUATIONS.                   *
C                 * * * * * * * * * * * * * * * * * * * * * * * * *
C
      IF(J.EQ.1.OR.M.EQ.1)GO TO 450
      EX=EX1+(EX2*ABS(EX2)/(2.*G*COS(SLOS)))
      U=(-G*COS(SLOS))*SIGN(1.,EX)
      GO TO 500
  450 EX=EX1+(EX2*ABS(EX2)/(2.*G*COS(SIGT)))
```

```
      U=(-G*COS(SIGT))*SIGN(1.,EX)
C
C                * * * * * * * * * * * * * * * * * * * * * * * * * *
C                * STATE VARIABLE EQLATIONS                          *
C                * * * * * * * * * * * * * * * * * * * * * * * * * *
C
  50C XDOT(1)=X(2)
      XCOT(2)=U
C
C                * * * * * * * * * * * * * * * * * * * * * * * * * *
C                * DETERMINE MISSILE VELOCITIES AND ACCELERATIONS.  *
C                * THE BRANCHING LOGIC FREVENTS POSSIBLE DIVISION    *
C                * BY ZERO. HOWEVER, THE VALUE OF X(2) SHCULD        *
C                * NEVER EXCEED THE MISSILE'S VELOCITY. IF THIS      *
C                * CONDITICN OCCURRED, THE RESULTS WOULC BE          *
C                * INVALID.                                          *
C                * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      IF(ABS(X(2)).LT.VM)GO TO 600
      VPX=0.
      VMXD=0.
      GO TO 700
  600 VPX=SQRT(VM*VM-X(2)*X(2))
      VMXD=(-U)*X(2)/VMX
  700 VMY=X(2)
      VPYD=U
C
C                * * * * * * * * * * * * * * * * * * * * * * * * * *
C                * CALCULATION OF ANGLE OF MISSILE FROM REFERENCE   *
C                * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      SIGM=ARSIN(X(1)/RM)
C
C                * * * * * * * * * * * * * * * * * * * * * * * * * *
C                * CALCULATION OF RATE OF CHANGE IN THE MISSILE'S   *
C                * ANGULAR POSITION.                                 *
C                * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      SGMD=(+1./SQRT(1.-X(1)*X(1)/(RM*RM)))*(X(2)/RM-X(1)/(RM*T))
C
C                * * * * * * * * * * * * * * * * * * * * * * * * * *
C                * MISSILE Y POSITION CETERMINED                     *
C                * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(7)=RM*SIN(SIGM)
C
```

```
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * MISSILE X POSITION DETERMINED                    *
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(8)=RM*COS(SIGM)
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * REVISE THE VALUES OF X7 AND X8 SO THEY WILL NOT *
C                  * ALTER THE VALUE OF X(7) AND X(8), THE MISSILE'S *
C                  * POSITION, DURING THE NEXT ITERATION.            *
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X7=X(7)
      X8=X(8)
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * STORE CURRENT VALUES OF X(1) ANC X(2).          *
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      W=X(1)
      WW=X(2)
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * FREEZE PROGRAM AFTER 9.9 SECONDS. THE MISSILE   *
C                  * IS ASSUMED TO GO BALLISTIC AT THIS TIME.        *
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      IF(T.GE.9.9)C(1)=2.
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * FREEZE PROGRAM, AFTER 8 SECONCS OF FLIGHT,      *
C                  * IF BOTH TIME AND MISSILE-TO-TARGET RANGE ARE    *
C                  * GREATER THAN THE PREVIOUS ITERATION.            *
C                  * MISSILE HAS REACHED CPA.                        *
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      IF(X(12).GT.Z.AND.T.GT.TT.AND.T.GT.8.0)C(1)=2.
C
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C                  * STORE CURRENT VALUES OF TIME AND MISSILE-TO-    *
C                  * TARGET RANGE FOR USE DURING THE NEXT ITERATION. *
C                  * VALUES ARE USED TO DETERMINE IF CPA REACHED.    *
C                  * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      TT=T
      Z=X(12)
C
```

```
C                 * * * * * * * * * * * * * * * * * * * * * * * * * *
C                 * VARIABLES PREPARED FOR OUTPUT                   *
C                 * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      X(3)=VMX
      X(4)=VMXD
      X(5)=VIX
      X(6)=VIY
      X(9)=SIGM*180./PI
      X(10)=U
      X(11)=SIGT*180./PI
      X(13)=E*180./PI
      X(14)=EDOT*180./PI
      X(15)=TOFR
      X(16)=VI
      X(19)=SLOS*180./PI
      X(20)=EX1
      X(21)=EX2
      X(22)=J
      X(23)=SIGTD*180./PI
      X(24)=SLOSD*180./PI
      X(25)=M
      X(26)=AC
      X(29)=TOFRD
      GO TO 1
      END
C***** INSERT A CARD WITH /* IN COLUMNS ONE AND TWO AT THIS POSITION
//GO.FT06F001 DD SPACE=(CYL,(6,1))
//GO.SYSIN DD *
1.        3.
5.        6.
6.6       7.0
7.4       7.8
8.2       8.6
 CONSOLIDATED SIMULATION
4
 2
1.0        .005       10.

150.      150.
TIME    00CRE     20CRERATE 21M-T SEP 12MX POSIT08TX POSIT28MY POSIT07TY POSIT27
 CRE VS TIME    2000 CRE RATE VS CRE212C
1.0        .005       10.

150.      150.
TIME    00SIGM    09SIGT    11SLOS    15SIGTD   23SLOSD   24MANEUVER25JAMMING 22
POSITION VS TIME0800                0700                2800                2700
1.0        .005       10.
```

```
150.      150.
TIME    00AC      26TOFR    15TOFRC   29MX RATE C3MY RATE C2MX ACCEL04MY ACCEL30
RELATIVE POSITS 0708                2728
1.0       .005        10.

150.      150.
TIME    00E       13EDOT    14IX POSIT18IY POSIT17IX RATE 05IY RATE 06IV      16
C***** INSERT /* ORANGE CARD AT THIS POSITION IN THE PROGRAM
```

## SUBROUTINE INTEG2

NOTE, INTEG1/2 SUBROUTINES PRODUCE THE SAME RESULTS. HOWEVER, INTEG2 PRODUCES PLOTTED OUTPUT ON THE VERSATEC PLOTTER WHILE INTEG1 PRODUCES PLOTTEC OUTPUT ON THE REGULAR OFF-LINE PRINTER.

```
      SUBROUTINE INTEG1(/TC/,/XC/,/DX/,/C/)                             INT12560
      REAL*8 ITITLE(12),JTITLE(8),KTITLE(8),IBLANK/'        '/          INT12570
      DIMENSION X(30),BX(30),XC(30),C(15),IP(10),IG(10),PR(10),GR(10),  INT12580
     1TX(5),TY(5),X1(900),Y1(900),X2(900),Y2(9C0),X3(900),Y3(900),      INT12590
     2X4(900),Y4(900)                                                   INT12600
      REAL LABEL,RUN(2)/' RUN','    '/,BT(9)/' 1  ',' 2  ',' 3  ',' 4  'INT12610
     1,' 5  ',' 6  ',' 7  ',' 8  ',' 9  '/                              INT12620
      ECUIVALENCE (ITITLE(7),RUN(1))                                    INT12630
      INDIC = C(10)+0.0000001                                           INT12640
      GO TO (1, 2000, 50, 58, 88, 88),INDIC                             INT12650
C                                                                       INT12660
C         READ DATA AND PRINT RECORD.                                   INT12670
C                                                                       INT12680
    1 READ(5,100) (ITITLE(I), I=1,6)                                    INT12690
  100 FCRMAT (10A8)                                                     INT12700
      READ (5,101)NR                                                    INT12710
  101 FORMAT (I1)                                                       INT12720
      READ (5,102)NN                                                    INT12730
  102 FORMAT (I2)                                                       INT12740
      NRC = 0                                                           INT12750
      IF(NN.LE.30) GO TO 1000                                           INT12760
      WRITE(6,200)                                                      INT12770
  200 FORMAT (////,48H ERROR IN ORDER OF EQUATION. MUST NOT EXCEED 30. )INT12780
      STOP                                                              INT12790
 1000 NRC = NRC + 1                                                     INT12800
      WRITE (6,201) (ITITLE(I),I=1,6)                                   INT12810
  201 FORMAT (1H1,/////,36X,6A8)                                        INT12820
      IF(NRC.EQ.1.AND.NR.EQ.1) GO TO 5                                  INT12830
      WRITE(6,202)NR                                                    INT12840
  202 FORMAT (/,37X,I1,20H RUNS ARE CALLED FOR )                        INT12850
      GO TO 6                                                           INT12860
    5 WRITE(6,203)                                                      INT12870
  203 FORMAT (/,37X,21HONE RUN IS CALLEC FOR ,///,18H INPUT DATA RECORD)INT12880
      GO TO 7                                                           INT12890
    6 WRITE(6,204)NRC                                                   INT12900
  204 FORMAT (///,34H INPUT DATA RECORD FOR RUN NUMBER    ,I1)          INT12910
    7 WRITE(6,205)NN                                                    INT12920
 205 FORMAT (//,22H ORDER OF EQUATIONS = ,I2)                           INT12930
     READ (5,103)TI,DT,TF1,DT2,TF2,DT3,TF3                              INT12940
 103 FORMAT (8F10.4)                                                    INT12950
     TF = TF1                                                           INT12960
     IF(DT2.NE.0.) GO TO 9                                              INT12970
     WRITE(6,206)TI,TF                                                  INT12980
 206 FORMAT (22H INITIAL TIME         = ,E10.4,/                        INT12990
    1        22H FINAL TIME           = ,E10.4)                         INT13000
     WRITE(6,207)DT                                                     INT13010
 207 FORMAT (22H STEP SIZE            = ,E10.4)                         INT13020
     GO TO 12                                                           INT13030
   9 IF(DT3.NE.0.) GO TO 11                                             INT13040
     TF = TF2                                                           INT13050
     WRITE(6,206) TI,TF                                                 INT13060
     WRITE(6,208)DT,TI,TF1,DT2,TF1,TF                                   INT13070
 208 FORMAT (22H STEP SIZE            = ,E10.4,13H BETWEEN T = ,E10.4,  INT13080
    1        9H AND T = ,E10.4)                                         INT13090
     GO TO 12                                                           INT13100
  11 TF=TF3                                                             INT13110
     WRITE(6,208) DT,TI,TF1,DT2,TF1,TF2,DT3,TF2,TF                      INT13120
  12 READ(5,103) (C(I),I=1,8)                                           INT13130
     READ (5,103)(X(I),I=1,NN)                                          INT13140
     J = 0                                                              INT13150
     DC 14 I=1,8                                                        INT13160
     IF(C(I).NE.0.) J=J+1                                               INT13170
  14 CONTINUE                                                           INT13180
     K = 0                                                              INT13190
     DO 16 I=1,NN                                                       INT13200
     IF(X(I).NE.0.) K=K+1                                               INT13210
  16 CONTINUE                                                           INT13220
     IF(J - 1)17,18,19                                                  INT13230
  17 WRITE (6,209)                                                      INT13240
 209 FORMAT (/,34H ALL THE CONSTANTS, C(I), ARE ZERO )                  INT13250
     GO TO 423                                                          INT13260
  18 WRITE (6,210)                                                      INT13270
 210 FORMAT (/,30H THE ONLY NON-ZERO CCNSTANT IS )                      INT13280
     GO TO 420                                                          INT13290
  19 WRITE (6,211)                                                      INT13300
 211 FORMAT (/,35H THE NON-ZERO CONSTANTS, C(I), ARE )                  INT13310
 420 DO 422 I=1,8                                                       INT13320
     IF(C(I).NE.0.) WRITE(6,212) I,C(I)                                 INT13330
 212 FORMAT (14X,2HC(,I2,4H) = ,E10.4)                                  INT13340
 422 CONTINUE                                                           INT13350
 423 IF(K - 1)424,425,426                                               INT13360
 424 WRITE (6,1209)                                                     INT13370
1209 FCRMAT (/,36H ALL THE INITIAL CONDITIONS ARE ZERO )                INT13380
     GO TO 20                                                           INT13390
 425 WRITE (6,1210)                                                     INT13400
 1210 FORMAT (/,35H THE ONLY NON-ZERO INITIAL CONCITION IS )
      GO TO 427
  426 WRITE (6,1211)
 1211 FORMAT (/,36H THE NON-ZERO INITIAL CONDITIONS ARE )
  427 DO 429 I=1,NN
      IF(X(I).NE.C.) WRITE(6,1212) I,X(I)
 1212 FORMAT (14X,2HX(,I2,4H) = ,E10.4)
  429 CONTINUE
   20 READ (5,104) (JTITLE(I),IP(I),I=1,8)
  104 FORMAT(8(A8,I2))
C
C     CHECK FOR THE NUMBER OF COLUMNS CALLED FOR BY LOCATING FIRST
C     BLANK COLUMN HEADING
C
      DO 21 J=1,8
      IF(JTITLE(J).EQ.IBLANK) GO TO 22
   21 CONTINUE
      J = 9
   22 JJ = J - 1
C
C     JJ IS NOW THE NUMBER OF COLUMNS. REPEAT WITH THE GRAPHS.
C
      READ (5,105)(KTITLE(I),KTITLE(I+1),IG(I),IG(I+1),I=1,7,2)
  105 FORMAT (4(2A8,2I2))
      DC 24 K=1,7,2
      IF(KTITLE(K).EQ.IBLANK.AND.KTITLE(K+1).EQ.IBLANK) GO TO 25
   24 CCNTINUE
      K = 8
   25 KK = K/2
      KKK = KK*2
      MLLTIP = 0
      IF(KK.NE.1) GO TO 306
      IF(IG(3) + IG(4).EQ.0) GO TO 306
      IF(IG(5) + IG(6).NE.0) GO TO 303
      MULTIP = 2
      KKK = 4
      GO TO 306
  303 IF(IG(7) + IG(8).NE.0) GO TO 305
      MLLTIP = 3
      KKK = 6
      GO TO 306
  3C5 MULTIP = 4
      KKK = 8
C
C     IF MULTIP = 0, KK IS THE NUMBER OF SINGLE CURVE GRAPHS. OTHERWISE
C     MULTIP IS THE NUMBER OF CURVES ON A SINGLE GRAPH.
C
  3C6 IF(JJ.EQ.0) GO TO 27
```

```
      WRITE(6,214) (JTITLE(I),IP(I),I=1,JJ)
  214 FORMAT (///,56H THE COLUMN HEADINGS AND THE CORRESPONDING VARIABLE
     1S ARE ,//,(10X,A8,4X,2HX(,I2,1H)))
      GO TO 28
   27 WRITE(6,215)
  215 FORMAT (///,25H NC PRINTOUT IS REQUIRED   )
   28 IF(KK.EQ.0) GO TO 308
      IF(MULTIP.NE.0) GO TO 309
      IF(KK.NE.1) GO TO 307
      WRITE (6,216) KTITLE(1),KTITLE(2),IG(1),IG(2)
  216 FORMAT (///,52H THE GRAPH TITLE AND THE CORRESPONDING VARIABLES AR
     1E ,//,10X,2A8,4X,2HX(,I2,8H) VS. X(,I2,1H))
      GO TO 31
  307 WRITE (6,217) (KTITLE(I),KTITLE(I+1),IG(I),IG(I+1),I=1,KKK,2)
  217 FORMAT (///,64H THE INDIVIDUAL GRAPH TITLES AND THE CORRESPONCING
     1VARIABLES ARE ,//,(10X,2A8,4X,2HX(,I2,8H) VS. X(,I2,1H)))
      GO TO 31
  308 WRITE (6,1217)
 1217 FORMAT(///,24H NO GRAPHS ARE REQUIRED   )
      GO TO 31
  309 WRITE(6,1220)
 1220 FORMAT (///,52H THE GRAPH TITLE AND THE CORRESPONDING VARIABLES AR
     1E ,/)
      WRITE(6,1221) KTITLE(1),KTITLE(2),(IG(I),IG(I+1),I=1,KKK,2)
 1221 FORMAT (10X,2A8,4X,2HX(,I2,8H) VS. X(,I2,1H),/, (30X,2HX(,I2,
     1           8H) VS. X(,I2,1H)))
C
C     THIS ENDS THE BOOK-KEEPING. INITIALIZE BEFORE ENTERING MAIN LOOP.
C
   31 IPAGE = 0
      T = TI
      NOPTS = 0
      NUMPTS = 0
      ITITLE(8)=IBLANK
      ITITLE(11)=IBLANK
      ITITLE(12)=IBLANK
      RUN(2)=BT(NRC)
      C(11) = 20.
      C(12) = 5.
      C(13) = DT
      DO 42 I=1,NN
   42 XC(I) = X(I)
      TC = T
      C(10) = 2.
      RETURN
C
 2000 IF(JJ.EQ.0) GO TO 54
C
```

```
      INCPR = C(11)+0.0000001                                             INT14370
      C(11) = 20.                                                         INT14380
      IF( MOD (NOPTS,50*INCPR).EQ.0) GO TO 46                             INT14390
      IF( MOD (NOPTS,10*INCPR).EQ.0) GC TO 47                             INT14400
      IF( MOD (NOPTS,    INCPR))54,48,54                                  INT14410
   46 IPAGE = IPAGE + 1                                                   INT14420
      IF(NR.EQ.1) GO TO 1047                                              INT14430
      WRITE (6,218) (ITITLE(I),I=1,6),IPAGE,ITITLE(7),(JTITLE(I),I=1,8)   INT14440
      WRITE(6,219)                                                        INT14450
      GO TO 47                                                            INT14460
 1047 WRITE (6,1218)(ITITLE(I),I=1,6),IPAGE,(JTITLE(I),I=1,8)             INT14470
      WRITE (6,219)                                                       INT14480
   47 WRITE(6,219)                                                        INT14490
  218 FCRMAT (1H1,///,20X,6A8,10X,5HPAGE ,I1,14H OF OUTPUT FOR,A8,//////,INT14500
     1           11X,8(A8,5X))                                            INT14510
 1218 FORMAT (1H1,///,20X,6A8,30X,5HPAGE ,I1,/////,11X,8(A8,5X))          INT14520
  219 FORMAT (1H  )                                                       INT14530
   48 DO 49 I=1,NN                                                        INT14540
   49 XC(I) = X(I)                                                        INT14550
      TC = T                                                              INT14560
      C(10) = 3.                                                          INT14570
      RETURN                                                              INT14580
C                                                                         INT14590
   50 DO 53 I=1,JJ                                                        INT14600
C                                                                         INT14610
      PR(I) = T                                                           INT14620
      IF(IP(I).NE.0) PR(I)=XC(IP(I))                                      INT14630
   53 CCNTINUE                                                            INT14640
      WRITE (6,22C)(PR(I),I=1,JJ)                                         INT14650
  220 FORMAT (7X, 8E13.5)                                                 INT14660
   54 IF(KK.EC.0) GO TO 62                                                INT14670
      INCGR = C(12)+0.0000001                                             INT14680
      C(12) = 5.                                                          INT14690
      IF( MOD (NOPTS, INCGR).NE.0) GO TO 62                               INT14700
      DO 57 I=1,NN                                                        INT14710
   57 XC(I) = X(I)                                                        INT14720
      TC = T                                                              INT14730
      C(10) = 4.                                                          INT14740
      RETURN                                                              INT14750
C                                                                         INT14760
   58 DO 61 I=1,KKK                                                       INT14770
C                                                                         INT1478C
      GR(I) = T                                                           INT14790
      IF(IG(I).NE.0) GR(I)=XC(IG(I))                                      INT14800
   61 CONTINUE                                                            INT14810
      IF (KKK .GE. 8) GO TO 1610                                          INT14820
      KP1 = KKK + 1                                                       INT14830
      DC 1612 I=KP1,8                                                     INT1484C
 1612 GR(I) = 0.                                                        INT14850
 1610 NUMPTS = NUMPTS + 1                                               INT14860
      Y1(NUMPTS) = GR(1)                                                INT14870
      X1(NUMPTS) = GR(2)                                                INT14880
      Y2(NUMPTS) = GR(3)                                                INT14890
      X2(NUMPTS) = GR(4)                                                INT14900
      Y3(NUMPTS) = GR(5)                                                INT14910
      X3(NUMPTS) = GR(6)                                                INT14920
      Y4(NUMPTS) = GR(7)                                                INT14930
      X4(NUMPTS) = GR(8)                                                INT14940
   62 NOPTS = NOPTS + 1                                                 INT14950
      IF(NUMPTS.LT.900) GO TO 64                                        INT14960
      WRITE (6,221)                                                     INT14970
  221 FCRMAT (//////,25H STOP AT 900 GRAPH POINTS    )                  INT14980
      GO TO 91                                                          INT14990
   64 IF(NOPTS.LT.4500) GO TO 66                                        INT15000
      WRITE (6,222)                                                     INT1501C
  222 FORMAT (//////,31H STOP AT 4500 INTEGRATION STEPS    )            INT15020
      GO TO 91                                                          INT15030
   66 IF(IPAGE - 9)69,67,68                                             INT15040
   67 IF( MOD (NOPTS , 50*INCPR).NE.0) GO TO 69                         INT15050
   68 WRITE (6,223)                                                     INT15060
  223 FORMAT (//////,27H STOP AT 9 PAGES OF OUTPUT   )                  INT1507C
      GO TO 91                                                          INT15080
   69 DC 70 I=1,NN                                                      INT15090
      IF(ABS(X(I)).GT.1.E+12) GO TO 71                                  INT15100
   70 CONTINUE                                                          INT15110
      GC TO 72                                                          INT15120
   71 WRITE (6,224)                                                     INT15130
  224 FORMAT (//////,76H STOP WITH THE ABSOLUTE VALUE OF A DEPENDENT VARINT15140
     1IABLE GREATER THAN 1.0E+12.   ,/,57H INTEGRATION PROBABLY UNSTABLE.INT15150
     2 TRY A SMALLER STEP SIZE. ,26HNO GRAPHS WILL BE PLCTTED.        ) INT15160
      GO TO 330                                                         INT15170
   72 DT = C(13)                                                        INT15180
      IF(TI.GT.TF) GO TO 80                                             INT15190
   73 IF(T.LT.TF) GO TO 75                                              INT15200
   74 WRITE (6,225)                                                     INT15210
  225 FORMAT (//////,26H NORMAL STOP AT FINAL TIME    )                 INT15220
      GC TO 91                                                          INT15230
   75 IF(T.GE.TF1) GO TO 77                                             INT15240
   76 C(13) = DT                                                        INT15250
      GO TO 87                                                          INT15260
   77 IF(T.GE.TF2) GO TO 79                                             INT15270
   78 C(13) = DT2                                                       INT15280
      GO TO 87                                                          INT15290
   79 C(13) = DT3                                                       INT15300
      GO TO 87                                                          INT15310
   80 IF(TF.GE.T) GO TO 74                                              INT15320
```

```
      IF(TF1.LT.T) GO TO 76
      IF(TF2 - T)78,79,79
   87 C(10) = 5.
C
   88 CALL RKUTTA (NN,T,X,DT,C,TC,XC,DX)
C
      IF(C(10).EQ.6.) RETURN
   90 T = T + DT
      GO TO 2000
   91 IF(KK.EQ.0) GO TO 330
      IF(MULTIP.NE.0) GO TO 97
C
C     PRINT PLOT LP TO 4 INDIVIDUAL CURVES
C
      NUMPTS=-NUMPTS
      DO 310 II=1,KK
      WRITE(6,9998)
 9998 FORMAT(1H1)
      ITITLE(9)=KTITLE(2*II-1)
      ITITLE(10)=KTITLE(2*II)
      GO TO (311,312,313,314),II
  311 CALL PLOTP(X1,Y1,NUMPTS,0)
      GO TO 310
  312 CALL PLOTP(X2,Y2,NUMPTS,0)
      GO TO 310
  313 CALL PLOTP(X3,Y3,NUMPTS,0)
      GO TO 310
  314 CALL PLOTP(X4,Y4,NUMPTS,0)
  310 WRITE(6,9999) ITITLE
 9999 FORMAT(1H0,8X,12A8)
      GO TO 330
C
C     PLOT DUMMY CURVE ALONG AXES TO SET SCALES FOR MULTIPLE PLOT
C
   97 BIGX = 0.
      BIGY = 0.
      SMLX = 0.
      SMLY = 0.
      DO 1970 I=1,NUMPTS
      XMAX =AMAX1  ( X1(I), X2(I), X3(I), X4(I))
      YMAX =AMAX1  ( Y1(I), Y2(I), Y3(I), Y4(I))
      XMIN =AMIN1  (X1(I), X2(I), X3(I), X4(I))
      YMIN =AMIN1 ( Y1(I), Y2(I), Y3(I), Y4(I))
      IF(BIGX.LT.XMAX) BIGX=XMAX
      IF(BIGY.LT.YMAX) BIGY=YMAX
      IF(SMLX.GT.XMIN) SMLX=XMIN
      IF(SMLY.GT.YMIN) SMLY=YMIN
 1970 CONTINUE
```

```
      TX(1) = 0.                                                        INT15810
      TX(2) = 0.                                                        INT15820
      TX(3) = 0.                                                        INT15830
      TX(4) = SMLX                                                      INT15840
      TX(5) = BIGX                                                      INT15850
      TY(1) = BIGY                                                      INT15860
      TY(2) = SMLY                                                      INT15870
      TY(3) = 0.                                                        INT15880
      TY(4) = 0.                                                        INT15890
      TY(5) = 0.                                                        INT15900
      WRITE(6,9998)                                                     INT15910
      ITITLE(9) = KTITLE(1)                                             INT15920
      ITITLE(10)= KTITLE(2)                                             INT15930
      NT=-5                                                             INT15940
      CALL  PLOTP(TX,TY,NT,1)                                           INT15950
      MODCUR   =  2                                                     INT15960
      DO 410  II=1,MULTIP                                               INT15970
      IF(II.EQ.MULTIP)  MODCUR=3                                        INT15980
      GO  TO    (411,412,413,414),II                                    INT15990
  411 CALL  PLOTP(X1,Y1,NUMPTS,MODCUR)                                  INT16000
      GO TO 410                                                         INT16010
  412 CALL  PLOTP(X2,Y2,NUMPTS,MODCUR)                                  INT16020
      GC TO 410                                                         INT16030
  413 CALL  PLOTP(X3,Y3,NUMPTS,MODCUR)                                  INT16040
      GC TO 410                                                         INT16050
  414 CALL  PLOTP(X4,Y4,NUMPTS,MODCUR)                                  INT16060
  410 CONTINUE                                                          INT16070
      WRITE(6,9999) ITITLE                                              INT16080
C                                                                       INT16090
  330 IF(NRC.NE.NR) GO TO 1000                                          INT16100
      IF(NR.GT.1) GO TO 333                                             INT16110
      WRITE(6,226)                                                      INT16120
  226 FORMAT(//,43H THE ONE RUN CALLED FOR HAS BEEN COMPLETED. ,//)     INT16130
      STOP                                                              INT16140
  333 WRITE(6,227)NR                                                    INT16150
  227 FORMAT(//,5H THE ,I1,37H RUNS CALLED FOR HAVE BEEN COMPLETED.,//) INT16160
      STOP                                                              INT16170
      END                                                               INT16180
C
C
C
C
C
```

```
      SUBROUTINE RKUTTA(/NN/,/T/,/X/,/DT/,/C/,/TC/,/XC/,/DX/)          INT16190
      DIMENSION X(30), C(15), XC(30), DX(30), CT(4), AK(4,30)          INT16200
      REAL*8 AK,CT                                                      INT16210
      INDIC = C(10) - 4.0+0.0000001                                     INT16220
      IF(INDIC.GT.1) GO TO 3                                            INT16230
      CT(1) = 0.0D0                                                     INT16240
      CT(2) = 0.5D0                                                     INT16250
      CT(3) = 0.5D0                                                     INT16260
      CT(4) = 1.0D0                                                     INT16270
      II=0                                                              INT16280
    7 II=II+1                                                           INT16290
      TC = T + CT(II)*DT                                                INT16300
      DO 2 J=1,NN                                                       INT16310
    2 XC(J) = X(J) + CT(II)*AK(II-1, J)                                 INT16320
      C(10) = 6.0                                                       INT16330
      RETURN                                                            INT16340
    3 DO 4 J=1,NN                                                       INT16350
    4 AK(II, J) = DT*DX(J)                                              INT16360
      IF(II.LT.4) GO TO 7                                               INT16370
      DO 5 J=1,NN                                                       INT16380
    5 X(J)=X(J)+(AK(1,J)+2.0*(AK(2,J)+AK(3,J))+AK(4,J))/6.0            INT16390
      C(10) = 7.0                                                       INT16400
      RETURN                                                            INT16410
      END                                                               INT16420
```

# BIBLIOGRAPHY

1. U.S. Army Foreign Science and Technology Center Report FSTC-1202-75, The Roland-1 Armored Self-Propelled AA Missile System, by Johannes and Weyand, p. 9-20, 10 June 1976.

2. Gibson, J. E., Nonlinear Automatic Control, p. 237, 342, 439-449, McGraw-Hill, 1963.

3. Goodstein, R., "Guidance Law Applicability for Missile Closing," Guidance and Control of Tactical Missiles, AGARD Lecture Series No. 52, 1972.

4. Heap, E., "Methodology of Research into Command-to-Line-of-Sight and Homing Guidance," Guidance and Control of Tactical Missiles, AGARD Lecture Series No. 52, 1972.

5. Ogata, K., Modern Control Engineering, p. 153-154, 537-538, 563-605, 663-704, Prentice-Hall, 1970.

## INITIAL DISTRIBUTION LIST

| | | No. Copies |
|---|---|---|
| 1. | Defense Documentation Center<br>Cameron Station<br>Alexandria, Virginia 22314 | 2 |
| 2. | Library, Code 0142<br>Naval Postgraduate School<br>Monterey, California | 2 |
| 3. | Chairman, EW Academic Group, Code 73<br>Naval Postgraduate School<br>Monterey, California 93940 | 1 |
| 4. | Professor H. A. Titus, Code 62Ts<br>Department of Electrical Engineering<br>Naval Postgraduate School<br>Monterey, California 93940 | 10 |
| 5. | LCDR Frank F. Hewitt<br>2315 Geranium Street<br>San Diego, California 92109 | 1 |
| 6. | Mr. Larry D. Parks and<br>Mr. Ron Russell<br>ADTC/ADA<br>Eglin AFB, Florida 32542 | 1 |
| 7. | Mr. Jack Willett<br>Stanford Research Institute<br>333 Ravenswood Avenue<br>Menlo Park, California 94025 | 1 |
| 8. | Mr. David Powell<br>DRDMI-YPE<br>U.S. Army Missile R & D Command<br>Redstone Arsenal, AL 35809 | 1 |
| 9. | Mr. Arlo A. Herrick<br>3246th TW TEER<br>Eglin AFB, Florida 32542 | 1 |
| 10. | Mr. Stan Wagner, Code 3513<br>Naval Weapons Center<br>China Lake, California 93555 | 1 |
| 11. | Mrs. Sylvia L. Waller<br>Headquarters USAF<br>AF/PACC<br>The Pentagon<br>Washington, D. C. 20330 | 1 |

No. Copies

| | | |
|---|---|---|
| 12. | Mr. Vern Green<br>Naval Air Systems Command<br>AIR 53322E<br>Washington, D. C.  20361 | 1 |